MiniSeg: An Extremely Minimum Network based on Lightweight Multiscale Learning for Efficient COVID-19 Segmentation

Yu Qiu, Yun Liu✉, Shijie Li, Jing Xu✉

***Abstract*—The rapid spread of the new pandemic, *i.e.,* COVID-19, has severely threatened global health. Deep-learning-based computer-aided screening, *e.g.,* COVID-19 infected area segmentation from CT image, has attracted much attention by serving as an adjunct to increase the accuracy of COVID-19 screening and clinical diagnosis. Although lesion segmentation is a hot topic, traditional deep learning methods are usually data-hungry with millions of parameters, easy to overfit under limited available COVID-19 training data. On the other hand, fast training/testing and low computational cost are also necessary for quick deployment and development of COVID-19 screening systems, but traditional methods are usually computationally intensive. To address the above two problems, we propose MiniSeg, a lightweight model for efficient COVID-19 segmentation from CT images. Our efforts start with the design of an Attentive Hierarchical Spatial Pyramid (AHSP) module for lightweight, efficient, effective multiscale learning that is essential for image segmentation. Then, we build a two-path encoder for deep feature extraction, where one path uses AHSP modules for learning multiscale contextual features and the other is a shallow convolutional path for capturing fine details. The two paths interact with each other for learning effective representations. Based on the extracted features, a simple decoder is added for COVID-19 segmentation. For comparing MiniSeg to previous methods, we build a comprehensive COVID-19 segmentation benchmark. Extensive experiments demonstrate that the proposed MiniSeg achieves better accuracy because its only 83K parameters make it less prone to overfitting. Its high efficiency also makes it easy to deploy and develop. The code has released at https://github.com/yun-liu/MiniSeg.**

***Index Terms*—COVID-19 segmentation, lightweight networks, computer-aided COVID-19 screening, lesion segmentation.**

I. Introduction

As one of the most severe pandemics in human history, *coronavirus disease 2019* (COVID-19) have caused serious damage to the global medical system, education, economy, and society. In order to control the spread of COVID-19, almost all countries have formulated various policies, *i.e.,* closing schools and factories, requiring people to wear masks and get the COVID-19 vaccination [1]. Moreover, lots of recent studies aim at developing the artificial intelligence based models to predict and control the casualties of COVID-19 [1]–[4]. However, due to the special characteristics of COVID-19, including rapid spread speed, high infectious, and strong insidiousness during the early stage, COVID-19 still threatens global health with thousands of newly infected patients every day. Hence, effective screening of infected patients is of high importance to the fight against COVID-19. The gold standard for COVID-19 diagnosis is the tried-and-true Reverse Transcription Polymerase Chain Reaction (RT-PCR) testing [5]. Unfortunately, the sensitivity of RT-PCR testing is not high enough to prevent the spread of COVID-19 [6]–[11]. Thus, the radiological image, *e.g.,* X-ray, computed tomography (CT), and ultrasound, can be used as a complementary tool for RT-PCR testing to improve screening sensitivity [6]–[8], [11]–[13]. Besides, radiological image analysis is necessary for clinical monitoring of disease severity [14]. However, radiological image examination needs expert radiologists, but we severely lack experienced radiologists during this pandemic. Therefore, computer-aided systems are expected for automatic radiological image interpretation.

When it comes to computer-aided COVID-19 screening, deep-learning-based technology is a good choice due to its uncountable successful stories in diagnosing, detecting or segmenting the infected areas from the radiological image [15]–[22]. For example, some works [23], [24] construct COVID-19 diagnosis systems using deep neural networks, which classify the CT or X-ray images into the categories such as normal, pneumonia, and COVID-19. However, their COVID-19 diagnosis can not accurately locate and segment the exact infected areas of COVID-19. In this paper, we aim at the COVID-19 segmentation which can provide more useful information for COVID-19 diagnosis. In fact, lots of recent studies focus on improving the performance of COVID-19 segmentation based on deep learning technology. For example, Fan *et al.* [25] proposed a novel COVID-19 lung infection segmentation network (Inf-Net) to automatically identify infected regions from CT images. Chen *et al.* [26] introduced a residual attention U-Net for automated three-class segmentation of COVID-19 CT images, including ground class opacification, consolidation, and pleural effusion. Amyar *et al.* [27] presented a multitask deep learning model to jointly identify COVID-19 patient and segment COVID-19 infected areas from CT images. They trained the neural network with three tasks, *i.e.,* reconstruction, classification, and segmentation, which were jointly performed with different datasets.

However, directly applying traditional deep learning models for COVID-19 segmentation is suboptimal. On the one hand, these models usually have millions of parameters and thus re-

Yu Qiu and Jing Xu are with College of Artificial Intelligence, Nankai University, Tianjin 300350, China.

Yun Liu is with Computer Vision Lab, Department of Information Technology and Electrical Engineering, ETH Zurich, Zurich 8092, Switzerland.

Shijie Li is with Department of Information Systems and Artificial Intelligence, University of Bonn, Bonn 53115, Germany.

Yun Liu and Jing Xu are joint corresponding authors.

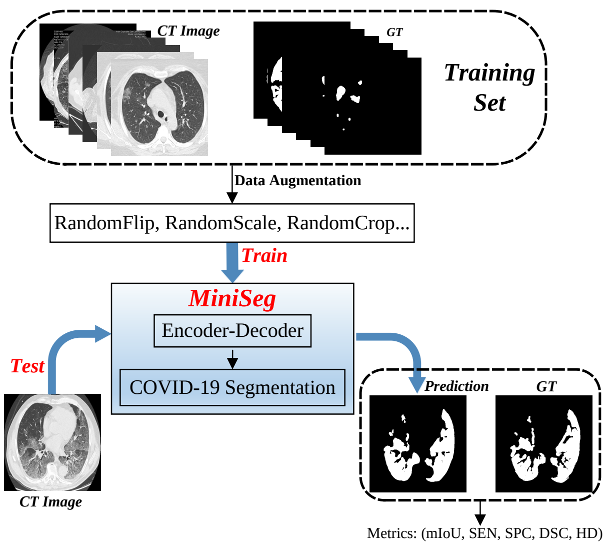

Fig. 1. Block diagram of the system.

quire a large amount of labeled data for training. The problem is that the publicly available COVID-19 segmentation data are limited and thus easy to cause overfitting for traditional data-hungry models [8]. To overcome the lack of labeled data, Fan *et al.* [25] further presented a semi-supervised segmentation framework based on a randomly selected propagation strategy, while the semi-supervised segmentation framework could not achieve sufficiently robust performance. On the other hand, these traditional deep learning methods, especially the ones for image segmentation, are usually computationally intensive. Considering the current severe pandemic situation, fast training/testing and low computational load are essential for quick deployment and development of computer-aided COVID-19 screening systems. Although Paluru *et al.* [11] tried to introduce a lightweight network for segmentation of anomalies in COVID-19 CT images, the efficiency was not enough to cope with the current COVID-19 situation.

It is a widely accepted concept that overfitting is easier to happen when a model has more parameters and less training data. It is also widely accepted that the overfitting can negatively impact the performance of the methods. To solve the above problems of COVID-19 segmentation, we observe that lightweight networks are not only uneasy to overfit owing to their small number of parameters but also likely to be efficient, making them suitable for computer-aided COVID-19 screening systems. Therefore, we think lightweight COVID-19 segmentation should be the technical solution of this paper. The key is to achieve accurate segmentation under the constraints of the number of network parameters and high efficiency. Although replacing the vanilla convolution with the combination of the *depthwise separable convolution* (**DSConv**) and *pointwise convolution* [28], [29] can reduce the number of network parameters, the accuracy usually decreases as the network shrinks [28]–[32]. To achieve a good trade-off between efficacy and efficiency, we observe that the accuracy of image segmentation can be improved with effective multiscale learning, which has significantly pushed forward the state of the arts of segmentation [33]–[41]. Hence, we resort to multiscale learning to ensure the segmentation accuracy of lightweight networks.

With the above analyses, our effort starts with the design of an **A**ttentive **H**ierarchical **S**patial **P**yramid (**AHSP**) module for lightweight, efficient, effective multiscale learning. AHSP first builds a spatial pyramid of dilated depthwise separable convolutions and feature pooling for learning multiscale semantic features. Then, the learned multiscale features are fused hierarchically to enhance the capacity of multiscale representation. Finally, the multiscale features are merged under the guidance of the attention mechanism, which learns to highlight essential information and filter out noisy information in radiography images. With the AHSP module incorporated, we propose an extremely minimum network, namely **MiniSeg**, for efficient segmentation of COVID-19 infected areas in chest CT slices. MiniSeg consists of a two-path encoder for deep feature extraction and a simple decoder for infected area segmentation. For the two-path encoder, one path uses AHSP modules for learning multiscale contextual information, and the other path is a shallow convolutional path for capturing fine details. The two paths interact with each other for learning effective, complementary feature representations. The block diagram of this system is shown in Fig. 1.

The proposed MiniSeg only has 83K parameters, two orders of magnitude less than traditional image segmentation methods, so that current limited COVID-19 data can be enough for training MiniSeg. At last, we build a comprehensive COVID-19 segmentation benchmark, including the well-known methods for both medical image segmentation and semantic image segmentation, to compare MiniSeg with previous methods. Compared to the preliminary version [42], this paper provides more details, discussion, and experiments to make it clearer and more convincing. Extensive experiments demonstrate that MiniSeg performs favorably against previous state-of-the-art segmentation methods with high efficiency.

In summary, our contributions are threefold:

- We propose an **A**ttentive **H**ierarchical **S**patial **P**yramid (**AHSP**) module for lightweight, efficient, effective multiscale learning that is essential for image segmentation.
- With AHSP incorporated, we present an extremely minimum network, **MiniSeg**, for accurate and efficient COVID-19 segmentation with limited training data.
- For an extensive comparison of MiniSeg with previous state-of-the-art segmentation methods, we build a comprehensive COVID-19 segmentation benchmark.

II. Related Work

In this section, we briefly review recent development in image segmentation and lightweight network design. We also discuss some recent studies for computer-aided COVID-19 screening and current public COVID-19 imaging datasets.

A. Image Segmentation

Image segmentation is a hot topic due to its wide range of applications. Since the invention of *fully convolutional networks* (FCNs) [54], FCN-based methods have dominated

TABLE I
A SUMMARY OF PUBLIC COVID-19 IMAGING DATASETS.

#Num	Dataset	Modality	#Total/#COVID	#Patients	Classification	Segmentation
1	COVID-19 Image Data Collection [43]	X-ray	542/434	262	COVID/Others	✗
2	COVID-CT-Dataset [44]	CT slice	912/349	216	COVID/Not	✗
3	SIRM COVID-19 Database [45]	X-ray & CT slice	384/384	71	COVID	✗
4	BIMCV COVID-19+ [46]	X-ray & CT slice	5381/2428	1311	COVID/Not/Others	✗
5	COVID-19 Radiography Database [47]	X-ray	21165/3616	-	COVID/Not/Others	✗
6	Lung Ultrasound (POCUS) Dataset [48]	Ultrasound image	1103/654	64	COVID/Not/Others	✗
7	SARS-CoV-2 CT-scan Dataset [49]	CT slice	2482/1252	-	COVID/Not	✗
8	COVID-19 X-rays [50]	X-ray	1442/224	-	COVID/Not/Others	✗
9	COVID-19 CT Segmentation Dataset [51]	CT slice	100/100	$\sim$60	COVID	✔
10	COVID-19 CT Segmentation Dataset [51]	CT slice	829/373	9	COVID	✔
11	COVID-19 CT Lung and Infection Segmentation Dataset [52]	CT slice	1844/1844	20	COVID	✔
12	MosMedData [53]	CT slice	785/785	1110	COVID/Not	✔

* "#Total" and "#COVID" denote the numbers of all or COVID-19 infected X-rays/CT slices, respectively. "#Patients" indicates the number of patients.

this field. Multiscale learning plays an essential role in image segmentation because objects in images usually exhibit very large scale variations. Hence, most current state-of-the-art methods aim at designing FCNs to learn effective multiscale representations from input images. In this direction, a popular way is to aggregate multiscale deep features from multi-level network layers for final dense prediction. For example, Ronneberger *et al.* [55] proposed the well-known U-Net that is actually an encoder-decoder network for fusing deep features from the top to bottom layers. U-Net++ [56] improves U-Net by introducing a series of nested, dense skip connections between the encoder and decoder sub-networks. Attention U-Net [57] improves U-Net by using the attention mechanism to learn to focus on target structures. DeconvNet [58] and SegNet [59] also make careful designs to improve the U-Net architecture. Moreover, some studies also try to extract multiscale features directly. For example, DeepLab [33] and its variants [34]–[36] design ASPP modules using dilated convolutions with different dilation rates to learn multiscale features. Based on ASPP, DenseASPP [36] connects a set of dilated convolutional layers densely, such that it generates multiscale features that cover a larger scale range densely.

Besides the multiscale learning, some studies focus on exploiting the global context information through pyramid pooling [37], context encoding [60], or non-local operations [61], [62]. Moreover, DFN [41] introduces a smooth network to handle the intra-class inconsistency problem and a border network to make the bilateral features of boundary distinguishable. Wu *et al.* [63] tried to find a good compromise between network depth and width to improve segmentation accuracy. The above models aim at improving segmentation accuracy without considering the model size and inference speed, so they are suboptimal for COVID-19 segmentation that only has limited training data and requires high efficiency. In the experiment section, we will show that the limited COVID-19 training data cannot optimize these large models well.

B. Lightweight Networks

Lightweight networks aim at reducing the parameters and computational complexity of deep networks. Convolution factorization is an intuitive way towards this goal. Specifically, many well-known network architectures decompose the standard convolution into multiple steps to reduce the computational complexity, including Flattened Model [64], Inception networks [65], Xception [28], ResNeXt [66], MobileNets [29], [30], and ShuffleNets [31], [32]. Among them, Xception [28] and MobileNets [29], [30] factorize a convolution into a pointwise convolution and a depthwise separable convolution. The pointwise convolution is actually a 1×1 convolution, which is used for interaction among channels. The depthwise separable convolution is a grouped convolution with the number of groups equaling to the number of output channels, so that it can process each feature channel separably. ShuffleNets [31], [32] further factorize a pointwise convolution into a channel shuffle operation and a grouped pointwise convolution for reducing the parameters and complexity. There are also some studies focusing on efficient semantic segmentation network design [38], [39], [67]–[69]. ESPNet [38] decomposes the standard convolution into a pointwise convolution and a spatial pyramid of dilated convolutions. ESPNetv2 [39] extends ESPNet [38] using grouped pointwise and dilated DSConv. Considering COVID-19 segmentation, our proposed MiniSeg should have a small number of parameters for training with limited data. Our observation of the essential role of multiscale learning in image segmentation helps MiniSeg achieve higher accuracy while being efficient.

Another way to build efficient networks is network compression. Previous studies have adopted various techniques, such as shrinking [70], parameter quantization [71], pruning [72], and hashing [73], to compress networks. Some research [74] also quantizes the network weights into low bits to reduce the model size and computational complexity. Moreover, some sparse methods are invented to reduce the redundancy of deep networks [75]–[77]. For example, Bagherinezhad *et al.* [75] encoded convolutions by a few lookups to a dictionary that is trained to cover the space of network weights. Liu *et al.* [76] proposed a sparse decomposition method, and Wen *et al.* [77] presented a structured sparsity learning method to regularize the network structures. However, these methods have to train deep networks before their compression. In this paper, we must avoid the training of large networks owing to the shortage of COVID-19 data, as discussed above. Therefore, these methods are orthogonal to our goal because they aim at compressing pretrained large networks rather than directly train a lightweight one.

C. Computer-aided COVID-19 Screening

Computer-aided COVID-19 screening has attracted much attention to serve as a supplementary tool for RT-PCR testing to improve screening sensitivity [6]–[13], [16], [17]. Some studies [8]–[10], [15], [78]–[83] design deep neural networks to classify chest X-rays or CT slices for computer-aided COVID-19 screening. In this paper, we focus on segmenting COVID-19 infected areas from chest CT slices because segmentation can provide more useful information than image classification. Fan *et al.* [25] proposed a segmentation model for COVID-19 infected area segmentation from CT slices. However, their method falls into the same category as previous segmentation methods, *i.e.,* with a large model size, and is thus suboptimal.

COVID-19 imaging datasets are the basis for computer-aided COVID-19 screening. Some public datasets have been introduced such as COVID-19 Image Data Collection [43], COVID-CT-Dataset [44], SIRM COVID-19 Database [45], BIMCV COVID-19+ [46], COVID-19 Radiography Database [47], Lung Ultrasound (POCUS) Dataset [48], SARS-CoV-2 CT-scan Dataset [49], COVID-19 X-rays [50], COVID-19 CT Segmentation Dataset [51], COVID-19 CT Lung and Infection Segmentation Dataset [52], and MosMedData [53], as summarized in Table I. We can see that most datasets only have image-level labels, and only four datasets [51]–[53] provide pixel-wise labels for COVID-19 infected area segmentation. Note that COVID-19 CT Segmentation Dataset [51] has two versions. Hence, we use these four segmentation datasets in our study.

III. Methodology

In this section, we first introduce the **A**ttentive **H**ierarchical **S**patial **P**yramid (**AHSP**) module for effective and lightweight multiscale learning. Then, we present the network architecture of MiniSeg for COVID-19 infected area segmentation from chest CT slices.

A. Attentive Hierarchical Spatial Pyramid Module

Although the factorization of a convolution operation into a pointwise convolution and a depthwise separable convolution (**DSConv**) can significantly reduce the number of network parameters and computational complexity, it usually comes with the degradation of accuracy [29]–[32]. Inspired by the fact that effective multiscale learning plays an essential role in improving segmentation accuracy [33]–[41], we propose the AHSP module for effective and efficient multiscale learning in a lightweight setting. Besides some common convolution operations, such as vanilla convolution, pointwise convolution, and DSConv, we introduce the dilated DSConv convolution that adopts a dilated convolution kernel for each input channel. Suppose $\mathcal{F}_r^{k\times k}$ denotes a vanilla convolution, where $k \times k$ is the size of convolution kernel and r is the dilation rate. Suppose $\hat{\mathcal{F}}_r^{k\times k}$ denotes a depthwise separable convolution, where $k \times k$ and r have the same meaning as $\mathcal{F}_r^{k\times k}$. The

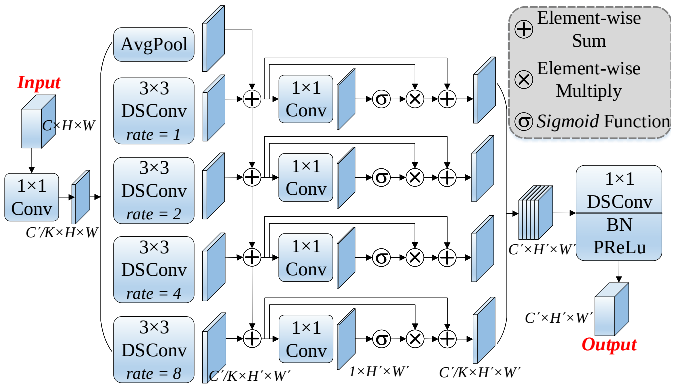

Fig. 2. Illustration of the proposed AHSP module.

subscript r will be omitted without ambiguity if we have a dilation rate of 1, *i.e.,* $r = 1$. For example, $\mathcal{F}^{1\times 1}$ represents a pointwise convolution (*i.e.,* 1×1 convolution). $\hat{\mathcal{F}}_2^{3\times 3}$ is a dilated 3×3 DSConv with a dilation rate of 2.

With the above definitions for basic operations, we continue by introducing the proposed AHSP module illustrated in Fig. 2. Let $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$ be the input feature map so that the output feature map is $\mathcal{E}(\mathbf{X}) \in \mathbb{R}^{C'\times H'\times W'}$, where $\mathcal{E}$ denotes the transformation function of AHSP for its input. C, H, and W are the number of channels, height, and width of the input feature map $\mathbf{X}$, respectively. Similar definitions hold for C', H', and W'. The input feature map $\mathbf{X}$ is first processed by a pointwise convolution to shrink the number of channels into C'/K, in which K is the number of parallel branches which will be described later. This operation can be written as

$$\mathbf{S} = \mathcal{F}^{1\times 1}(\mathbf{X}). \tag{1}$$

Then, the generated feature map $\mathbf{S}$ is fed into K parallel dilated DSConv, *i.e.,*

$$\mathbf{F}_k = \hat{\mathcal{F}}_{2^{k-1}}^{3\times 3}(\mathbf{S}), \quad k = 1, 2, \cdots, K, \tag{2}$$

where the dilation rate is increased exponentially for enlarging the receptive field. Eq. (2) is the basis for multiscale learning with large dilation rates capturing large-scale contextual information and small dilation rates capturing local information. We also add an average pooling operation for $\mathbf{S}$ to enrich the multiscale information, *i.e.,*

$$\mathbf{F}_0 = \mathrm{AvgPool}^{3\times 3}(\mathbf{S}), \tag{3}$$

where $\mathrm{AvgPool}^{3\times 3}$ represents the average pooling with a kernel size of 3×3. Note that we have $\mathbf{F}_k \in \mathbb{R}^{\frac{C'}{K}\times H'\times W'}$ for $k = 0, 1, \cdots, K$. If we have $H \neq H'$ or $W \neq W'$, the convolution and pooling operations in Eq. (2) and Eq. (3) will have a stride of 2 to downsample the feature map by a scale of 2; otherwise, the stride will be 1.

These multiscale feature maps produced by Eq. (2) and Eq. (3) are aggregated in an attentive hierarchical manner. We first add them up hierarchically as

$$\begin{aligned} \dot{\mathbf{F}}_1 &= \mathbf{F}_0 + \mathbf{F}_1, \\ \dot{\mathbf{F}}_2 &= \dot{\mathbf{F}}_1 + \mathbf{F}_2, \\ &\cdots \\ \dot{\mathbf{F}}_K &= \dot{\mathbf{F}}_{K-1} + \mathbf{F}_K, \end{aligned} \tag{4}$$

where feature maps are gradually fused from small scales to large scales to enhance the representation capability of multiscale learning. We further adopt a spatial attention mechanism to make the AHSP module automatically learn to focus on target structures of various scales. On the other hand, the attention mechanism can also learn to suppress irrelevant information at some feature scales and emphasize essential information at other scales. Such self-attention makes each scale speak for itself to decide how important it is in the multiscale learning process. The transformation of $\dot{\mathbf{F}}$ by spatial attention can be formulated as

$$\ddot{\mathbf{F}}_k = \dot{\mathbf{F}}_k + \dot{\mathbf{F}}_k \otimes \sigma(\mathcal{F}^{1\times1}(\dot{\mathbf{F}}_k)), \quad k = 1, 2, \cdots, K, \tag{5}$$

in which σ is a *sigmoid* activation function and $\otimes$ indicates element-wise multiplication. The pointwise convolution in Eq. (5) outputs a single-channel feature map which is then transformed to a spatial attention map by the *sigmoid* function. This attention map is replicated to the same size as $\dot{\mathbf{F}}_k$, *i.e.,* $\frac{C'}{K} \times H' \times W'$, before element-wise multiplication. Considering the efficiency, we can compute the attention map for all K branches together, like

$$\mathbf{A} = \sigma(\mathcal{F}^{1\times1}(\mathrm{Concat}(\dot{\mathbf{F}}_1, \dot{\mathbf{F}}_2, \cdots, \dot{\mathbf{F}}_K))), \tag{6}$$

where $\mathrm{Concat}(\cdot)$ means to concatenate a series of feature maps along the channel dimension. The pointwise convolution in Eq. (6) is a K-grouped convolution with K output channels, so we have $\mathbf{A} \in \mathbb{R}^{K\times H'\times W'}$. Hence, we can rewrite Eq. (5) as

$$\ddot{\mathbf{F}}_k = \dot{\mathbf{F}}_k + \dot{\mathbf{F}}_k \otimes \mathbf{A}[k], \quad k = 1, 2, \cdots, K, \tag{7}$$

in which $\mathbf{A}[k]$ means the k-th channel of $\mathbf{A}$.

Finally, we merge and fuse the above hierarchical feature maps as

$$\begin{aligned} \ddot{\mathbf{F}} \quad &= \mathrm{Concat}(\ddot{\mathbf{F}}_1, \ddot{\mathbf{F}}_2, \cdots, \ddot{\mathbf{F}}_K), \\ \mathcal{E}(\mathbf{X}) &= \mathrm{PReLU}(\mathrm{BatchNorm}(\mathcal{F}^{1\times1}(\ddot{\mathbf{F}}))), \end{aligned} \tag{8}$$

where $\mathrm{BatchNorm}(\cdot)$ denotes the batch normalization and $\mathrm{PReLU}(\cdot)$ indicates PReLU (*i.e.,* Parametric ReLU) activation function [84]. The pointwise convolution in Eq. (8) is a K-grouped convolution with C' output channels, so that this pointwise convolution aims at fusing $\ddot{\mathbf{F}}_k$ ($k = 1, 2, \cdots, K$) separately, *i.e.,* adding interaction among channels for the depthwise convolutions in Eq. (2). The fusion among various feature scales is achieved through the first pointwise convolution (*i.e.,* Eq. (1)) in the subsequent AHSP module of MiniSeg and the hierarchical feature aggregation (*i.e.,* Eq. (4)). Such a design can reduce the number of convolution parameters in Eq. (8) by K times when compared with that using a vanilla pointwise convolution, *i.e.,* C'^2/K *vs.* C'^2.

Given an input feature map $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$, we can compute the output feature map $\mathcal{E}(\mathbf{X}) \in \mathbb{R}^{C'\times H'\times W'}$ of an AHSP module using Eq. (1) - Eq. (8). We can easily find that increasing K will reduce the number of AHSP parameters. Considering the balance between segmentation accuracy and efficiency, we set $K = 4$ in our experiments. The proposed AHSP module not only significantly reduces the number of parameters but also enables us to learn effective multiscale features so that we can adopt the limited COVID-19 data to train a high-quality segmenter.

B. Network Architecture

Here, we describe in detail the network architecture of the proposed lightweight COVID-19 segmenter, *i.e.,* MiniSeg. MiniSeg has an encoder-decoder structure. The encoder sub-network focuses on learning effective multiscale representations for the input image. The decoder sub-network gradually aggregates the representations at different levels of the encoder to predict COVID-19 infected areas. The network architecture of MiniSeg is displayed in Fig. 3.

1) Encoder sub-network: The encoder of MiniSeg uses AHSP as the basic module, consisting of two paths connected through a series of nested skip pathways. Suppose $\mathbf{I} \in \mathbb{R}^{3\times H\times W}$ denotes an input chest CT slice, where a grayscale CT slice is replicated three times to make its number of channels the same as color images. The input $\mathbf{I}$ is downsampled four times, resulting in four scales of $1/2$, $1/4$, $1/8$, and $1/16$, with four stages processing such four scales, respectively. Downsampling happens in the first block of each stage. Previous semantic image segmentation models usually only downsample images into $1/8$ scale [33], [36]–[38], [40], [41], [60]–[62], [67], [68], [85], however, in this paper, we downsample until the $1/16$ scale for enlarging the receptive field and reducing the computational complexity.

Suppose in the encoder sub-network we denote the output feature map of the i-th stage and the j-th block as $\mathbf{E}^i_j$, *w.r.t.* $i \in \{1, 2, 3, 4\}$ and $j \in \{1, 2, \cdots, N_i\}$, where N_i indicates the number of blocks at the i-th stage. Therefore, we have $\mathbf{E}^i_j \in \mathbb{R}^{C_i\times\frac{H}{2^i}\times\frac{W}{2^i}}$, in which C_i is the number of feature channels at the i-th stage. The abovementioned block refers to the proposed AHSP module except for the first stage whose basic block is the vanilla **Convolution Block** (**CB**). Since the number of feature channels at the first stage (*i.e.,* C_1) is small, the vanilla convolution will not introduce too many parameters. Without ambiguity, let $\mathcal{E}^i_j(\cdot)$ be the transformation function of the i-th stage and the j-th block without distinguishing whether this block is a vanilla convolution or an AHSP module. For the another path, we propose a **Downsampler Block** (**DB**). The transformation function of a DB block is denoted as $\mathcal{Q}^i_k(\cdot)$, *w.r.t.* $i \in \{1, 2, 3, 4\}$ and $k \in \{1, 2, \cdots, M_i\}$, where M_i denotes the number of DB at the i-th stage. We define DB as

$$\mathcal{Q}^i_k(\mathbf{X}) = \mathrm{PReLU}(\mathrm{BatchNorm}(\hat{\mathcal{F}}^{5\times5}(\mathcal{F}^{1\times1}(\mathbf{X})))), \tag{9}$$

where $\hat{\mathcal{F}}^{5\times5}(\cdot)$ has a stride of 2 for downsampling when we have $k = 1$. Suppose the output of $\mathcal{Q}^i_k(\cdot)$ is $\mathbf{Q}^i_k$.

Therefore, for the first block of the first stage, we have

$$\mathbf{E}^1_1 = \mathcal{E}^1_1(\mathbf{I}), \quad \mathbf{Q}^1_1 = \mathcal{Q}^1_1(\mathbf{I}). \tag{10}$$

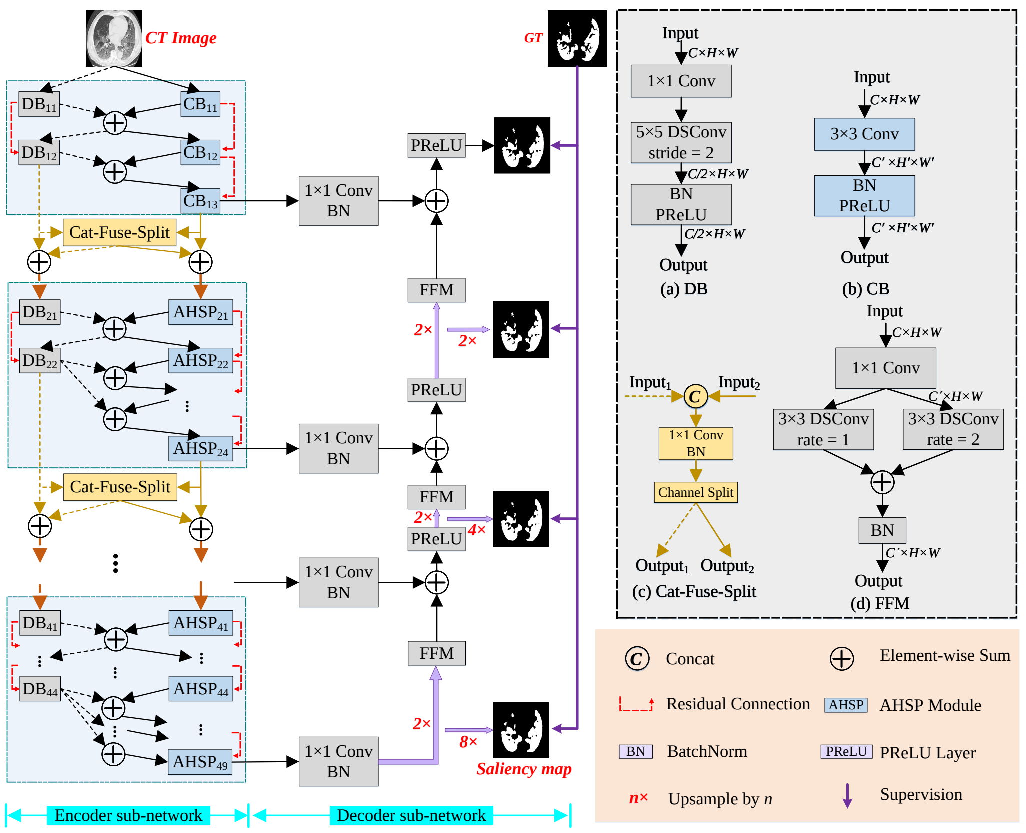

Fig. 3. Network architecture of the proposed MiniSeg.

For the first block of other stages, we compute the output feature map as

$$\begin{aligned}\mathbf{E}^{i-1} &= \mathcal{F}^{1\times1}(\mathrm{Concat}(\mathbf{E}^{i-1}_{N_{i-1}}, \mathbf{Q}^{i-1}_{M_{i-1}})),\\ \mathbf{E}^{i}_{1} &= \mathcal{E}^{i}_{1}(\mathrm{Split}(\mathbf{E}^{i-1}) + \mathbf{E}^{i-1}_{N_{i-1}}),\\ \mathbf{Q}^{i}_{1} &= \mathcal{Q}^{i}_{1}(\mathrm{Split}(\mathbf{E}^{i-1}) + \mathbf{Q}^{i-1}_{M_{i-1}}),\end{aligned} \tag{11}$$

where we have $i \in \{2,3,4\}$. The operation $\mathrm{Split}(\cdot)$ is to split a feature map along the channel dimension into two chunks, which are fed into $\mathcal{E}^i_1$ and $\mathcal{Q}^i_1$, respectively. Here, $\mathcal{E}^i_1(\cdot)$ and $\mathcal{Q}^i_1(\cdot)$ $(i \in \{1,2,3,4\})$ have a stride of 2 for downsampling. Instead of only using on-the-fly element-wise sum (Eq. (12) and Eq. (13)), through Eq. (11), we conduct a "concat-fuse-split" operation to fully integrate the features from the two paths, as concatenation can do better for feature fusion than sum by avoiding the information loss of sum [86]. $\mathrm{Split}(\cdot)$ is used to handle the increased number of channels brought by concatenation.

For other blocks, the output feature map is

$$\begin{aligned}\mathbf{E}^{i}_{j} &= \mathcal{E}^{i}_{j}(\mathbf{E}^{i}_{j-1} + \mathbf{Q}^{i}_{j'}) + \mathbf{E}^{i}_{j-1},\\ &\textit{w.r.t. } i \in \{1,2,3,4\} \text{ and } j \in \{2,3,\cdots,N_i\},\end{aligned} \tag{12}$$

where $\mathcal{E}^i_j(\cdot)$ has a stride of 1 and a residual connection is included for better optimization. We have $j' = j-1$ if we also have $j-1 \le M_i$; otherwise, we have $j' = M_i$. The computation of $\mathbf{Q}^i_k$ can be formulated as

$$\begin{aligned}\mathbf{Q}^{i}_{k} &= \mathcal{Q}^{i}_{k}(\mathbf{Q}^{i}_{k-1} + \mathbf{E}^{i}_{k-1}) + \mathbf{Q}^{i}_{k-1},\\ &\textit{w.r.t. } i \in \{1,2,3,4\} \text{ and } k \in \{2,3,\cdots,M_i\}.\end{aligned} \tag{13}$$

Through Eq. (12) and Eq. (13), the two paths of the encoder sub-network build nested skip connections. Such a design benefits the multiscale learning of the encoder. Considering the balance among the number of network parameters, segmentation accuracy, and efficiency, we set C_i to $\{8,24,32,64\}$, N_i to $\{3,4,9,7\}$, and M_i to $\{2,2,5,4\}$ for $i \in \{1,2,3,4\}$, respectively.

2) The decoder sub-network: The decoder of MiniSeg is simple for efficient multiscale feature decoding. Since the top feature map of the encoder has a scale of $1/16$ of the original input, it is suboptimal to predict COVID-19 infected areas directly due to the loss of fine details. Instead, we utilize a simple decoder sub-network to gradually upsample and fuse the learned feature map at each scale. A **Feature Fusion Module** (**FFM**) is proposed for feature aggregation. Let $\mathcal{D}_i(\cdot)$ represent the function of FFM:

$$\begin{aligned}\mathbf{S}'_i &= \mathcal{F}^{1\times1}(\mathbf{X}),\\ \mathcal{D}_i(\mathbf{X}) &= \mathrm{BatchNorm}(\hat{\mathcal{F}}^{3\times3}_1(\mathbf{S}'_i) + \hat{\mathcal{F}}^{3\times3}_2(\mathbf{S}'_i)),\end{aligned} \tag{14}$$

in which $\mathcal{D}_i(\mathbf{X})$ $(i = 1,2,3)$ has C_i channels as the pointwise convolution is utilized to adjust such number of channels. We denote the feature map in the decoder as $\mathbf{D}_i \in \mathbb{R}^{C_i \times \frac{H}{2^i} \times \frac{W}{2^i}}$, and we have $\mathbf{D}_4 = \mathrm{BatchNorm}(\mathcal{F}^{1\times1}(\mathbf{E}^4_{N_4}))$. We compute other $\mathbf{D}_i$ $(i = 3,2,1)$ as

$$\begin{aligned}\mathbf{S}''_i &= \mathcal{D}_i(\mathrm{Upsample}(\mathbf{D}_{i+1}, 2)),\\ \mathbf{D}_i &= \mathrm{PReLU}(\mathbf{S}''_i + \mathrm{BatchNorm}(\mathcal{F}^{1\times1}(\mathbf{E}^i_{N_i}))),\end{aligned} \tag{15}$$

where $\mathrm{Upsample}(\cdot, t)$ means to upsample a feature map by a scale of t using bilinear interpolation. In this way, the decoder sub-network enhances the high-level semantic features with low-level fine details, so that MiniSeg can make accurate predictions for COVID-19 infected areas.

3) Training and testing: With $\mathbf{D}_i$ $(i = 1, 2, 3, 4)$ computed, we can make dense prediction using a pointwise convolution, *i.e.,*

$$\mathbf{P}_i = \mathrm{Softmax}(\mathrm{Upsample}(\mathcal{F}^{1\times1}(\mathbf{D}_i), 2^i)), \tag{16}$$

where $\mathrm{Softmax}(\cdot)$ is the standard *softmax* function and this pointwise convolution has two output channels representing two classes of background and COVID-19, respectively. $\mathbf{P}_i \in \mathbb{R}^{H\times W}$ is the predicted class label map. In testing, we utilize $\mathbf{P}_1$ as the final output prediction.

In training, we replace the *softmax* function in Eq. (16) with the standard cross-entropy loss function, which can be formulated as

$$L_i = \mathcal{L}_{\mathrm{CEL}}(\mathrm{Upsample}(\mathcal{F}^{1\times1}(\mathbf{D}_i), 2^i), \mathbf{G}), \tag{17}$$

in which $\mathcal{L}_{\mathrm{CEL}}(\cdot)$ indicates the standard cross-entropy loss function and $\mathbf{G}$ is the ground-truth label map. We adopt deep supervision during training, so the total loss can be calculated as

$$L = L_1 + \lambda \cdot \sum_{i=2}^{4} L_i, \tag{18}$$

where λ is a balance weight. In this paper, we follow previous studies [37], [60], [70] to empirically set λ as 0.4.

IV. Experiments

A. Experimental Setup

1) Implementation details: We implement the proposed MiniSeg network using the well-known PyTorch framework [87]. Adam optimization [88] is used for training with the weight decay of 1e-4. We adopt the learning rate policy of *poly*, where the initial learning rate is 1e-3. We train 80 epochs on the training set with a batch size of 5. We train all previous state-of-the-art segmentation methods using the same training settings as MiniSeg for a fair comparison. All experiments are performed on a TITAN RTX GPU.

2) Datasets: As described in Section II, we utilize four open-access CT datasets for COVID-19 segmentation, *i.e.,* two versions of the COVID-19 CT Segmentation Dataset [51], COVID-19 CT Lung and Infection Segmentation Dataset [52], and MosMedData [53], to evaluate the proposed MiniSeg. According to the number of CT slices or the number of COVID-19 patients, we rename these four datasets as COVID-19-CT100, COVID-19-P9, COVID-19-P20, and COVID-19-P1110 for convenience, respectively. The information of these datasets is summarized in Table I. Specifically, COVID-19-CT100 [51] consists of 100 axial CT slices from ∼60 patients infected by COVID-19. COVID-19-P9 [51] consists of 9 axial volumetric CT scans with 829 slices in total, where 373 slices have been evaluated by a radiologist as positive. COVID-19-P20 [52] contains 1844 labeled axial CT slices from 20 COVID-19 patients. The last dataset COVID-19-P1110 [53] are from 1110 patients but only 785 CT slices from 50 patients has been annotated. We utilize the standard cropping and random flipping for data augmentation for MiniSeg and all baselines in training. Moreover, we perform **5-fold cross-validation** to avoid statistically significant differences in performance evaluation.

3) Evaluation metrics: We evaluate COVID-19 segmentation accuracy using five popular evaluation metrics in medical imaging analysis, *i.e.,* mean intersection over union (mIoU), sensitivity (SEN), specificity (SPC), Dice similarity coefficient (DSC), and Hausdorff distance (HD). The metric of mIoU is a typical measure for semantic segmentation by computing the overlap rate between prediction and ground truth for each class and then averaging across all classes. Here, sensitivity represents the probability of the COVID-19 infected area of ground truth to be predicted as it is. Specificity represents the probability of the background region of ground truth to be predicted as background. Dice similarity coefficient is an overlap index that can represent the degree of similarity between predicted COVID-19 area and ground-truth COVID-19 area. Hausdorff distance (HD) measures the structural differences among two given objects and is the minimum distance between the ground truth and segmented region. These metrics are defined as follows:

$$\mathrm{SEN} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}, \tag{19}$$

$$\mathrm{SPC} = \frac{\mathrm{TN}}{\mathrm{TN}+\mathrm{FP}}, \tag{20}$$

$$\mathrm{DSC} = \frac{2\times\mathrm{TP}}{2\times\mathrm{TP}+\mathrm{FP}+\mathrm{FN}}, \tag{21}$$

$$\mathrm{HD} = \max(h(S_m, S_a), h(S_a, S_m)), \tag{22}$$

where TP, FP, TN, FN indicate the number of pixels in the true positive, false positive, true negative, and false negative regions, respectively. $S_m = \{s_{m1}, s_{m2}, \ldots, s_{mi}\}$ is the curve generated from ground truth of the COVID-19 infected area, and $S_a = \{s_{a1}, s_{a2}, \ldots, s_{ai}\}$ is the curve formed by segmentation methods. Suppose we have $h(S_m, S_a) = \max_{s_m\in S_m} \min_{s_a\in S_a} ||s_m - s_a||$ where $||\cdot||$ is Euclidean distance; $h(S_a, S_m)$ can be similarly defined. Specifically, mIoU, SEN, SPC, and DSC range between 0 and 1. The larger these values, the better the model. Note that a lower value of HD indicates better segmentation accuracy. Moreover, we also report the number of parameters, the number of FLOPs, and speed, tested using a 512×512 input image and a TITAN RTX GPU.

B. Ablation Studies

Before comparing to other methods, we conduct ablation studies to demonstrate the effectiveness of our model components and design choices on four datasets.

TABLE II
EFFECT OF THE MAIN COMPONENTS IN MINISEG.

SB	MB	AH	TP	CS	Metrics (%) on COVID-19-CT00					Metrics (%) on COVID-19-P9					Metrics (%) on COVID-19-P20					Metrics (%) on COVID-19-P1110				
					mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓
✔					77.82	80.03	97.00	69.32	93.81	83.07	86.34	99.03	76.11	63.55	80.67	83.65	98.04	71.16	70.45	75.78	71.80	96.78	59.63	92.12
	✔				78.22	81.97	96.90	70.21	87.56	82.40	89.38	99.10	76.19	65.87	80.99	83.59	98.58	71.50	68.51	76.31	76.22	97.40	61.57	88.05
	✔	✔			79.63	83.33	97.07	72.30	89.24	83.50	90.39	**99.51**	76.96	83.42	81.63	**85.10**	98.25	71.72	66.38	76.58	77.89	97.61	62.06	83.71
	✔	✔	✔		80.80	84.87	97.38	73.81	81.79	83.88	89.58	99.28	77.61	78.17	82.77	83.75	98.20	73.78	57.00	76.66	78.72	97.02	62.05	78.67
	✔	✔	✔	✔	**82.15**	**84.95**	**97.72**	**75.91**	**74.42**	**85.31**	**90.60**	99.15	**80.06**	**58.46**	**84.49**	85.06	**99.05**	**76.27**	**51.06**	**78.33**	**79.62**	**97.71**	**64.84**	**71.69**

* A metric marked by ↑ means that a model is better if it achieves higher results in terms of this metric, while ↓ indicates the opposite. The HD metric does not have the unit of %.

TABLE III
EFFECT OF THE TWO-PATH DESIGN IN MINISEG.

Method	#Param ↓	FLOPs ↓	Speed ↑	Metrics (%) on CT100					Metrics (%) on P9					Metrics (%) on P9					Metrics (%) on P20				
				mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓
Single-path	472.44K	2.27G	232.6fps	81.92	**84.98**	96.59	75.08	**71.50**	82.86	85.79	**99.15**	75.32	64.28	83.23	84.70	98.57	74.81	53.44	77.14	79.03	97.21	62.73	76.29
Two-path	**82.91K**	**0.50G**	**516.3fps**	**82.15**	84.95	**97.72**	**75.91**	74.42	**85.31**	**90.60**	**99.15**	**80.06**	**58.46**	**84.49**	**85.06**	**99.05**	**76.27**	**51.06**	**78.33**	**79.62**	**97.71**	**64.84**	**71.69**

* #Param represents the number of network parameters.

TABLE IV
EFFECT OF SOME DESIGN CHOICES IN MINISEG.

PReLU	Decoder	DS	CB	5×5 DB	FFM	Metrics (%) on COVID-19-CT100					Metrics (%) on COVID-19-P9					Metrics (%) on COVID-19-P20					Metrics (%) on COVID-19-P1110				
						mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓
ReLU						80.56	84.59	97.33	72.04	82.76	81.43	**94.12**	98.85	73.28	**45.79**	82.81	80.17	96.51	71.52	54.45	76.92	75.41	96.90	62.11	78.39
	✗					78.77	83.97	96.83	70.67	82.86	79.05	87.55	98.08	69.97	73.24	82.04	**85.49**	98.47	72.27	54.86	73.11	76.31	97.45	55.35	76.72
		✗				80.74	84.38	97.41	74.13	78.52	81.62	85.66	**99.56**	73.68	80.54	82.67	85.09	98.75	73.95	58.97	76.45	**80.94**	96.38	62.46	87.27
			AHSP			80.83	**85.93**	97.32	74.51	78.56	81.51	86.51	99.31	74.15	75.02	82.71	84.04	98.69	73.98	56.10	76.71	78.38	96.38	62.05	78.99
				3×3 DB		80.42	85.81	97.18	73.43	86.78	82.43	87.50	98.75	74.69	85.44	82.34	84.03	98.56	73.72	54.72	76.54	78.43	97.09	61.81	80.61
					AHSP	80.45	85.24	97.17	73.44	79.50	81.01	85.85	99.01	73.74	68.83	82.84	83.78	98.03	73.82	57.00	77.15	78.69	97.33	62.46	82.98
						82.15	84.95	**97.72**	**75.91**	**74.42**	**85.31**	90.60	99.15	**80.06**	58.46	**84.49**	85.06	**99.05**	**76.27**	**51.06**	**78.33**	79.62	**97.71**	**64.84**	**71.69**

* Each design choice is replaced with the operation in the table or directly removed (✗). DS: Deep Supervision.

TABLE V
ABLATION STUDIES FOR THE MAIN PARAMETERS OF MINISEG.

Configurations		Metrics (%) on COVID-19-CT00					Metrics (%) on COVID-19-P9					Metrics (%) on COVID-19-P20					Metrics (%) on COVID-19-P1110				
		mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓
Default Configurations		**82.15**	84.95	**97.72**	**75.91**	**74.42**	**85.31**	**90.60**	99.15	**80.06**	**58.46**	**84.49**	85.06	**99.05**	**76.27**	51.06	**78.33**	79.62	**97.71**	**64.84**	**71.69**
$N_1 \sim N_4$	$\{4,5,10,8\}$	81.40	84.57	97.16	73.83	76.98	85.05	89.16	99.03	77.42	63.71	82.87	84.06	98.22	74.05	51.87	77.02	78.73	97.28	62.33	78.68
	$\{5,6,11,9\}$	81.58	84.73	97.30	74.38	77.41	84.84	88.85	99.05	78.53	61.04	83.04	**85.33**	98.49	74.90	54.71	76.83	79.12	96.80	62.96	74.56
	$\{2,3,8,6\}$	80.17	85.19	97.26	73.34	80.39	82.56	87.64	98.31	73.10	68.30	82.92	83.55	98.11	73.55	56.89	76.63	78.42	96.46	61.71	78.23
	$\{2,2,7,5\}$	80.33	84.87	97.23	73.39	80.55	83.21	88.01	98.87	74.89	80.50	82.65	83.89	98.62	74.01	58.27	76.91	77.85	96.96	62.51	78.66
$M_1 \sim M_4$	$\{3,3,6,5\}$	81.44	85.23	97.37	74.58	77.45	84.91	88.36	98.76	76.93	61.90	83.10	84.17	98.41	74.47	53.38	77.12	79.19	**97.71**	62.47	79.29
	$\{4,4,7,6\}$	81.12	84.53	97.20	73.86	76.57	85.01	88.23	98.91	77.49	65.05	82.94	84.78	98.56	74.45	54.25	77.03	79.65	97.20	62.82	80.57
	$\{2,2,4,3\}$	79.72	84.58	96.05	73.55	81.20	82.61	87.42	98.28	74.11	78.76	83.21	84.26	98.30	74.53	56.30	77.08	79.01	96.63	62.91	78.73
	$\{2,2,3,2\}$	79.85	84.95	96.05	74.20	81.50	82.34	87.74	99.10	74.94	104.83	82.77	84.28	98.00	74.04	52.02	76.87	79.37	97.25	62.63	75.04
$C_1 \sim C_4$	$\{16,32,64,128\}$	81.81	84.58	97.46	74.14	74.24	85.04	90.57	99.04	77.66	62.08	82.69	84.71	98.49	74.15	54.50	76.88	**80.20**	97.42	62.66	81.26
	$\{16,48,64,192\}$	81.24	82.94	97.71	74.47	75.72	84.88	87.37	98.76	76.75	64.55	83.34	85.08	98.69	75.13	**50.44**	77.46	79.86	97.58	63.36	79.77
	$\{8,16,32,64\}$	79.97	**85.33**	97.07	72.57	80.27	82.58	88.88	98.99	73.75	76.00	82.43	83.87	98.51	73.46	55.03	76.69	78.46	96.66	61.74	78.53
	$\{8,12,24,48\}$	79.97	83.99	97.09	72.25	79.61	82.39	86.89	98.86	73.13	78.26	82.15	83.49	98.75	73.40	57.78	76.61	79.02	97.34	61.70	79.66

* "N_i", "M_i", "C_i" indicate the number of AHSP/CB blocks, DB blocks, feature channels at the i-th stage for $i \in \{1,2,3,4\}$, respectively. Specially, we set N_i to $\{3,4,9,7\}$, M_i to $\{2,2,5,4\}$, and C_i to $\{8,24,32,64\}$ for $i \in \{1,2,3,4\}$ by default, respectively.

TABLE VI
COMPARISON OF MINISEG WITH DIFFERENT CONFIGURATIONS IN TERMS OF PARAMETERS, FLOPS, AND SPEED.

Configurations		#Param	FLOPs	Speed
Default Configurations		**82.91K**	**0.50G**	**516.3fps**
$N_1 \sim N_4$	$\{4,5,10,8\}$	87.92K	0.56G	464.6fps
	$\{5,6,11,9\}$	92.93K	0.61G	432.5fps
	$\{2,3,8,6\}$	77.90K	0.45G	571.2fps
	$\{2,2,7,5\}$	73.50K	0.43G	625.1fps
$M_1 \sim M_4$	$\{3,3,6,5\}$	92.26K	0.55G	460.6fps
	$\{4,4,7,6\}$	101.60K	0.59G	441.9fps
	$\{2,2,4,3\}$	75.10K	0.49G	518.0fps
	$\{2,2,3,2\}$	63.70K	0.47G	523.9fps
$C_1 \sim C_4$	$\{16,32,64,128\}$	262.87K	1.40G	312.4fps
	$\{16,48,64,192\}$	448.95K	1.88G	264.4fps
	$\{8,16,32,64\}$	78.61K	0.42G	567.1fps
	$\{8,12,24,48\}$	50.03K	0.33G	658.9fps

1) Effect of main components: As shown in Table II, we start with a **single-branch (SB)** module that only has the DSConv with a dilation rate of 1. Replacing all AHSP modules in MiniSeg with such SB modules and removing the two-path design of the MiniSeg encoder, we obtain the baseline of the proposed MiniSeg. As shown in the 1st line of Table II, this baseline achieves mIoU of 77.82%, 83.07%, 80.67%, and 75.78% on COVID-19-CT100, COVID-19-P9, COVID-19-P20, and COVID-19-P1110 datasets, respectively. Then, we extend such an SB module to a **multi-branch (MB)** module using the spatial pyramid as in the AHSP module. The results are summarized in the 2nd line of Table II. This MB module improves the SB module in almost all cases, indicating the necessity of multiscale learning in COVID-19 segmentation. Next, we add the **attentive hierarchical fusion strategy (AH)** to get the AHSP module, so we can improve the mIoU to 79.63%, 83.50%, 81.63%, and 76.58% on COVID-19-CT100, COVID-19-P9, COVID-19-P20, and COVID-19-P1110 datasets, respectively. Besides the mIoU metric, we can clearly see that the attentive hierarchical fusion strategy leads to further performance boost in most cases, as depicted in the 3rd line of Table II. This suggests the superiority of the attentive hierarchical fusion. We continue by adding the **two-path design (TP)** to the encoder sub-network. As shown in the 4th line of Table II, we further improve the mIoU to 80.80%, 83.88%, 82.77%, and 76.66% on COVID-19-CT100, COVID-19-P9, COVID-19-P20, and COVID-19-P1110 datasets, respectively. This validates that such a two-

TABLE VII
ABLATION STUDIES FOR THE HYPER-PARAMETERS OF MINISEG.

Configurations		Metrics (%) on COVID-19-CT00					Metrics (%) on COVID-19-P9					Metrics (%) on COVID-19-P20					Metrics (%) on COVID-19-P1110				
		mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓
Default Configurations		**82.15**	84.95	97.72	**75.91**	**74.42**	**85.31**	**90.60**	99.15	**80.06**	**58.46**	**84.49**	**85.06**	**99.05**	**76.27**	**51.06**	**78.33**	79.62	**97.71**	**64.84**	**71.69**
Learning Rate	5e-4	79.36	85.79	96.80	72.65	81.48	83.33	87.22	99.28	72.24	74.77	82.89	83.43	98.69	74.26	55.27	76.66	75.94	96.34	61.10	74.47
	1e-3	79.21	84.57	96.27	76.70	79.64	77.54	84.18	99.45	68.45	87.06	81.44	81.43	97.44	70.34	63.98	74.39	77.40	98.74	74.06	84.47
	5e-3	81.00	85.95	97.35	75.73	74.84	80.13	89.28	98.19	71.66	79.80	81.80	85.25	98.35	72.66	66.25	74.03	82.02	97.28	58.53	98.94
Batch Size	4	81.31	84.88	97.22	74.46	76.28	84.65	87.85	99.29	75.35	72.70	82.93	82.71	97.90	73.71	55.91	76.50	79.88	96.91	61.61	80.28
	6	81.14	84.74	97.08	73.64	83.22	831.72	88.46	98.74	74.06	84.47	82.91	84.29	98.77	74.16	57.30	76.69	79.35	97.42	62.63	82.11
	8	80.08	84.81	96.93	72.55	82.53	83.47	88.99	99.37	73.62	84.21	82.49	83.65	98.42	73.96	57.17	76.79	77.84	96.45	62.19	74.65
Epoch	50	79.08	84.37	95.74	71.93	76.46	82.34	87.15	98.99	71.73	89.61	82.62	83.73	98.78	73.38	57.37	76.40	80.43	97.78	61.89	81.22
	60	80.54	85.83	96.89	73.17	86.22	83.18	87.81	99.41	72.69	72.49	82.90	83.99	99.03	74.20	54.17	77.02	78.75	96.99	62.15	77.75
	100	81.70	85.84	97.21	74.25	75.80	85.13	84.75	98.57	74.34	70.27	82.78	84.32	98.52	74.24	55.76	77.30	78.31	96.64	62.88	77.54
Weight Decay	5e-5	80.71	84.82	95.99	73.18	84.52	82.11	90.15	99.27	74.80	69.50	82.97	84.96	98.72	74.44	53.73	76.46	78.19	96.72	62.13	77.34
	1e-5	80.54	**86.22**	96.93	73.27	82.39	81.78	85.80	99.33	73.32	68.33	82.96	84.67	98.63	74.22	52.23	76.93	77.15	97.19	62.58	76.74
	5e-4	80.30	85.17	96.12	73.78	82.15	81.08	87.66	98.40	72.80	84.54	83.08	87.15	98.77	74.44	57.80	75.91	79.60	97.09	60.97	88.63

* By default, the learning rate, batch size, number of training epochs, weight decay are set to 1e-4, 5, 80, 1e-4, respectively.

TABLE VIII
ABLATION STUDIES FOR THE RANDOM SEEDS DURING NETWORK TRAINING.

Random Seeds		Metrics (%) on COVID-19-CT00					Metrics (%) on COVID-19-P9					Metrics (%) on COVID-19-P20					Metrics (%) on COVID-19-P1110				
		mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓
Unfixed		82.15	84.95	97.72	75.91	74.42	85.31	90.60	99.15	80.06	58.46	84.49	85.06	99.05	76.27	51.06	78.33	79.62	97.71	64.84	71.69
Fixed	0	82.13	84.71	97.79	75.21	74.80	85.56	89.56	99.34	80.13	59.63	84.15	84.92	98.86	75.77	51.81	78.48	79.63	97.76	64.92	73.21
	1	82.33	85.19	97.66	75.21	74.85	85.08	89.18	99.28	78.85	60.85	84.51	85.13	99.00	76.79	54.03	78.02	79.15	97.43	64.59	72.64
	42	82.16	84.73	97.75	74.96	74.15	85.08	90.22	99.03	79.49	57.39	83.98	84.99	98.73	75.26	51.34	78.19	79.46	97.22	64.47	70.35
	100	82.00	85.55	97.83	75.26	74.64	85.77	88.54	98.93	80.31	56.17	84.37	85.05	98.90	76.18	52.29	78.30	79.64	97.80	65.01	72.16
Average Values		82.15	85.03	97.75	75.31	74.57	85.36	89.62	99.15	79.77	58.50	84.30	85.03	98.91	76.05	52.11	78.26	79.50	97.58	64.77	72.01
Standard Deviations		0.105	0.315	0.058	0.318	0.259	0.271	0.733	0.152	0.535	1.641	0.205	0.071	0.112	0.513	1.049	0.153	0.187	0.217	0.204	0.971

TABLE IX
COMPARISON BETWEEN MINISEG AND PREVIOUS STATE-OF-THE-ART SEGMENTATION METHODS.

Method	Metrics (%) on COVID-19-CT100					Metrics (%) on COVID-19-P9					Metrics (%) on COVID-19-P20					Metrics (%) on COVID-19-P1110				
	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓
U-Net [55]	77.56	72.24	97.71	68.37	94.25	76.51	88.53	98.93	65.69	133.64	81.81	82.73	97.92	72.66	61.66	74.26	**81.85**	97.33	58.62	95.72
FCN-8s [54]	71.85	66.47	93.56	58.11	104.68	81.20	87.12	98.40	72.67	91.32	82.54	84.10	98.02	73.60	51.47	70.51	80.75	97.08	53.33	84.43
SegNet [59]	75.02	80.02	96.34	64.84	109.05	73.88	73.59	98.79	62.07	98.38	79.55	81.68	98.44	69.68	77.28	72.32	76.77	97.24	55.92	105.42
FRRN [40]	79.20	78.47	97.50	71.27	86.56	80.83	86.26	99.54	74.03	84.34	80.61	80.75	97.53	71.43	61.28	73.84	75.45	95.80	58.86	87.11
PSPNet [37]	75.61	70.82	96.47	64.55	99.76	82.15	86.84	99.19	74.85	94.40	81.60	83.44	98.17	71.60	65.60	71.41	80.34	97.40	54.82	87.06
DeepLabv3 [34]	81.30	84.80	97.48	74.65	81.77	81.50	85.23	98.56	73.10	95.72	80.26	81.60	97.78	70.96	60.50	72.91	80.45	96.85	55.70	81.35
DenseASPP [36]	78.43	81.14	97.02	70.37	156.23	72.78	70.26	98.65	65.53	98.61	81.11	82.21	97.80	71.68	64.05	74.84	69.38	95.65	57.24	76.61
DFN [41]	81.07	84.27	97.49	74.45	83.73	79.19	85.78	98.64	69.93	106.23	79.13	80.96	96.51	69.46	66.56	73.40	80.12	97.13	57.31	87.10
EncNet [60]	71.28	74.11	95.20	62.83	119.55	81.35	86.88	98.65	72.62	94.77	82.43	84.94	98.03	71.60	71.57	71.65	81.23	96.65	54.89	77.82
DeepLabv3+ [35]	79.45	79.58	97.55	71.70	93.09	81.29	77.93	99.30	73.48	81.95	81.26	81.61	95.35	42.79	182.14	74.14	74.65	97.26	57.16	102.78
BiSeNet [89]	63.09	74.07	87.41	58.66	110.47	72.33	67.17	96.35	55.40	164.07	78.08	76.13	97.07	65.24	85.94	70.29	70.90	95.49	52.26	95.11
UNet++ [56]	77.64	77.26	97.28	69.04	91.73	77.95	86.83	99.39	69.27	104.83	80.73	79.61	96.75	70.34	63.01	73.39	75.67	96.13	59.08	88.21
Attention U-Net [57]	77.71	74.75	97.56	68.93	92.15	76.26	76.39	99.24	66.74	102.43	80.70	82.92	97.41	71.27	64.91	74.62	81.32	97.63	59.34	95.16
OCNet [90]	69.29	72.86	89.38	56.14	105.66	81.14	87.41	98.71	72.94	113.21	80.74	80.71	95.82	69.36	56.60	72.05	79.67	97.64	53.97	97.38
DUpsampling [91]	81.69	84.54	97.60	75.27	81.07	79.96	74.42	96.38	69.60	64.62	81.05	79.37	96.34	71.01	60.19	72.16	65.18	91.77	53.98	72.29
DANet [92]	73.57	66.30	92.76	61.34	99.11	81.59	88.78	99.13	73.82	114.69	78.35	79.87	97.31	67.04	83.13	73.47	75.00	95.80	56.07	74.04
CCNet [61]	75.24	69.55	95.92	63.99	98.03	81.27	86.61	99.16	73.93	90.84	82.22	82.93	97.76	73.13	56.98	72.02	79.16	96.29	54.83	83.07
ANNNet [62]	73.93	66.73	95.72	62.06	102.43	79.52	85.20	98.35	69.55	109.31	81.92	84.10	98.13	72.72	56.99	72.28	81.19	97.30	55.21	83.16
GFF [93]	75.75	69.80	97.53	63.88	103.87	81.20	85.35	98.46	72.61	113.48	82.44	84.29	97.49	73.05	63.84	71.82	81.10	96.50	53.88	86.39
Inf-Net [25]	81.62	76.50	**98.32**	74.44	86.81	80.28	77.59	98.72	71.76	69.46	64.62	69.46	99.02	63.38	79.68	74.32	62.93	93.45	56.39	71.77
MobileNet [29]	80.07	81.19	95.92	63.99	98.03	81.32	85.53	**99.62**	74.18	128.95	80.52	82.66	97.95	72.05	70.70	74.84	80.08	97.67	59.91	92.88
MobileNetv2 [30]	79.73	82.83	97.32	72.53	88.40	80.09	81.77	99.45	72.16	85.15	80.99	83.16	98.20	71.50	68.54	74.32	80.41	96.96	59.43	93.11
ShuffleNet [31]	77.50	74.57	97.64	69.02	86.97	80.87	83.62	99.28	72.66	105.56	81.97	82.34	98.03	73.33	56.68	74.51	77.73	96.38	58.64	78.16
ShuffleNetv2 [32]	78.58	81.21	97.30	71.37	84.72	79.54	82.44	98.75	70.29	102.75	81.31	81.86	98.29	71.67	70.06	74.56	76.89	96.58	58.67	78.55
EfficientNet [94]	78.22	80.25	97.04	70.45	75.26	73.13	73.50	99.25	60.20	133.45	81.58	80.10	98.06	72.12	64.30	73.30	80.66	97.07	58.04	96.30
ENet [85]	79.49	81.26	97.53	71.57	96.08	79.27	79.62	99.07	70.43	101.92	77.57	76.35	97.16	68.23	67.40	74.49	74.86	96.38	57.20	85.32
ESPNet [38]	77.45	84.18	96.48	69.30	97.04	76.79	71.30	98.67	67.68	93.58	80.32	80.53	97.52	69.36	91.84	74.75	72.06	96.96	57.77	94.58
CGNet [67]	79.34	81.55	96.34	71.42	90.37	75.10	70.27	92.57	60.37	134.43	82.24	80.73	97.35	72.35	53.63	74.12	74.83	96.16	56.45	74.34
ESPNetv2 [39]	78.66	77.84	96.53	70.46	87.77	78.22	72.42	97.23	67.12	88.58	80.78	79.03	97.41	70.13	73.67	74.10	76.60	97.67	58.37	96.73
EDANet [68]	78.74	82.86	96.98	70.67	88.14	80.11	79.40	98.77	72.89	70.40	79.56	76.79	97.42	68.71	70.72	73.21	73.73	96.71	55.11	84.56
LEDNet [69]	77.41	81.69	96.93	68.74	92.49	78.46	80.96	98.47	70.41	120.74	80.34	78.74	97.90	70.10	65.77	73.46	72.27	95.14	55.09	94.19
MiniSeg	**82.15**	**84.95**	97.72	**75.91**	**74.42**	**85.31**	**90.60**	99.15	**80.06**	**58.46**	**84.49**	**85.06**	**99.05**	**76.27**	**51.06**	**78.33**	79.62	**97.71**	**64.84**	**71.69**

path design can benefit the network optimization. At last, we add the **channel split (CS)** operation to obtain the final MiniSeg model, which further pushes the mIoU to 82.15%, 85.31%, 84.49%, and 78.33% on COVID-19-CT100, COVID-19-P9, COVID-19-P20, and COVID-19-P1110 datasets, respectively (the 5th line of Table II). These ablation studies demonstrate that the main components in MiniSeg are all effective for COVID-19 segmentation.

In Table II, we study the effect of the two-path design by evaluating only the main path (*i.e.,* $\mathbf{E}_j^i$). In Table III, we enlarge the number of channels of the single-path baseline so that it has the same channels as the concatenation of the two paths. From Table III, the single-path baseline has more parameters, more FLOPs, lower speed, and lower accuracy, suggesting the effectiveness of the two-path design.

TABLE X
COMPARISON OF MINISEG TO PREVIOUS STATE-OF-THE-ART METHODS IN TERMS OF PARAMETERS, FLOPS, AND SPEED.

Method	Publication	Backbone	ImageNet	#Param	FLOPs	Speed
U-Net [55]	MICCAI'2015	-	No	8.43M	65.73G	57.3fps
FCN-8s [54]	TPAMI'2017	VGG16	Yes	15.53M	105.97G	4.5fps
SegNet [59]	TPAMI'2017	-	No	28.75M	160.44G	3.0fps
FRRN [40]	CVPR'2017	-	No	17.30M	237.70G	15.8fps
PSPNet [37]	CVPR'2017	ResNet50	Yes	64.03M	257.79G	17.1fps
DeepLabv3 [34]	arXiv'2017	ResNet50	Yes	38.71M	163.83G	25.3fps
DenseASPP [36]	CVPR'2018	-	No	27.93M	122.28G	19.3fps
DFN [41]	CVPR'2018	ResNet50	Yes	43.53M	81.88G	56.2fps
EncNet [60]	CVPR'2018	ResNet50	Yes	51.25M	217.46G	18.1fps
DeepLabv3+ [35]	ECCV'2018	Xception	Yes	53.33M	82.87G	3.4fps
BiSeNet [89]	ECCV'2018	ResNet18	Yes	12.50M	13.01G	172.4fps
UNet++ [56]	DLMIA'2018	-	No	8.95M	138.37G	26.8fps
Attention U-Net [57]	MIDL'2018	-	No	8.52M	67.14G	49.2fps
OCNet [90]	IJCV'2021	ResNet50	Yes	51.60M	220.69G	19.3fps
DUpsampling [91]	CVPR'2019	ResNet50	Yes	28.46M	123.01G	36.5fps
DANet [92]	CVPR'2019	ResNet50	Yes	64.87M	275.72G	16.4fps
CCNet [61]	CVPR'2019	ResNet50	Yes	46.32M	197.92G	40.0fps
ANNNet [62]	ICCV'2019	ResNet50	Yes	47.42M	203.07G	32.8fps
GFF [93]	AAAI'2020	ResNet50	Yes	90.57M	374.03G	17.5fps
Inf-Net [25]	TMI'2020	ResNet50	Yes	30.19M	27.30G	155.9fps
MobileNet [29]	arXiv'2017	MobileNet	Yes	3.13M	3.02G	416.7fps
MobileNetv2 [30]	CVPR'2018	MobileNetv2	Yes	2.17M	1.60G	137.0fps
ShuffleNet [31]	CVPR'2018	ShuffleNet	Yes	0.92M	0.75G	116.3fps
ShuffleNetv2 [32]	ECCV'2018	ShuffleNetv2	Yes	1.22M	0.77G	142.9fps
EfficientNet [94]	ICML'2019	EfficientNet	No	8.37M	13.19G	48.1fps
ENet [85]	arXiv'2016	-	No	0.36M	1.92G	71.4fps
ESPNet [38]	ECCV'2018	-	No	0.35M	1.76G	125.0fps
CGNet [67]	TIP'2020	-	No	0.49M	3.40G	73.0fps
ESPNetv2 [39]	CVPR'2019	-	No	0.34M	0.77G	73.0fps
EDANet [68]	MM'2019	-	No	0.68M	4.43G	147.1fps
LEDNet [69]	ICIP'2019	-	No	2.26M	6.32G	94.3fps
MiniSeg	-	-	No	**82.91K**	**0.50G**	**516.3fps**

2) *Effect of some design choices:* Besides the above main components, we also conduct ablation studies for some design choices of MiniSeg. The results are provided in Table IV. First, we replace the PReLU activation function with the ReLU function. Second, we remove the decoder sub-network and change the stride of the last stage from 2 to 1, so we can directly make predictions at the scale of $1/8$ and upsample to the original size, as in previous studies [33], [36]–[38], [40], [41], [60], [61], [67], [68], [85]. Third, we remove deep supervision in training, which means that only L_1 in Eq. (18) is used. Fourth, we replace Convolution Blocks (CB) in the first stage with AHSP modules. Fifth, we replace the 5×5 DSConv in the Downsampler Blocks (DB) with 3×3 DSConv. Sixth, we replace the Feature Fusion Modules (FFM) in the decoder sub-network with AHSP modules. We can see that the default setting achieves the best overall performance on all four datasets, demonstrating the efficacy of our designs.

3) *Impact of main parameters:* In this part, we study the impact of various parameters of MiniSeg and explain how the default parameters are set. "N_i", "M_i", and "C_i" indicate the number of AHSP/CB blocks, DB blocks, feature channels at the i-th stage for $i \in \{1, 2, 3, 4\}$, respectively. Table V shows the segmentation results of MiniSeg with different configurations. In general, the default setting achieves the best performance. All other configurations can bring some performance degradation, but overall, MiniSeg is robust to various parameters. Moreover, Table VI displays the comparison of MiniSeg with different configurations in terms of parameters, FLOPs and speed. Considering the balance between segmentation accuracy and efficiency, we set the default number of AHSP/CB blocks, DB blocks, feature channels for four stages to $\{3, 4, 9, 7\}$, $\{2, 2, 5, 4\}$, and $\{8, 24, 32, 64\}$, respectively.

4) *Impact of training hyper-parameters:* In the following ablation studies, we provide justification for the choice of training hyper-parameters. We use the control variable method to explore the impact of the main training hyper-parameters, including the learning rate, batch size, training epochs, and weight decay. Table VII shows the performance of MiniSeg under different training settings. Considering a group of rows concerning a hyper-parameters of Table VII, we only change it and leave other hyper-parameters to their default values. In this way, the change of results in terms of five metrics under different values of a hyper-parameter can indicate the preference of this hyper-parameter. Following the recent study, we consider setting the learning rate, batch size, epochs and weight decay to (1e-4, 5e-5, 1e-3, 5e-3), (4, 5, 6, 8), (50, 60, 80, 100), and (1e-5, 5e-5, 1e-4, 5e-4), respectively. Specially, the default learning rate, batch size, epochs and weight decay are set to 1e-4, 5, 80, 1e-4, whose results are provided in the first row of Table VII. The results of other settings of these training hyper-parameters are listed in the rows of different groups in Table VII. From Table VII, we can conclude that MiniSeg can achieve the optimal performance under the default training hyper-parameters setting. Besides, the performance of MiniSeg under other training settings is still better than baseline models in general.

5) *Impact of random training seeds:* In our experiments, we train MiniSeg using 5-fold cross-validation with *unfixed* random seed to ensure the robustness of MiniSeg, and in fact, we only train our final model one time. In the following ablation studies, we train our MiniSeg multiple times with different random seeds to verify the impact of different random seeds and further compute the statistical measure, *i.e.,* standard deviations. Here, we select four widely-used random seeds, *i.e.,* 0, 1, 42, 100. From the results shown in Table VIII, we can conclude that the different random seeds only have little effect on segmentation performance, which is consistent to our viewpoint: the performance of a robust model should not rely on the value of random seeds.

C. Comparison with State-of-the-art Methods

1) *Quantitative Evaluation:* To compare MiniSeg to previous state-of-the-art competitors and promote COVID-19 segmentation research, we build a comprehensive benchmark. This benchmark contains 31 previous state-of-the-art image segmentation methods, including U-Net [55], FCN-8s [54], SegNet [59], FRRN [40], PSPNet [37], DeepLabv3 [34], DenseASPP [36], DFN [41], EncNet [60], DeepLabv3+ [35], BiSeNet [89], UNet++ [56], Attention U-Net [57], OCNet [90], DUpsampling [91], DANet [92], CCNet [61], ANNNet [62], GFF [93], Inf-Net [25], MobileNet [29], MobileNetv2 [30], ShuffleNet [31], ShuffleNetv2 [32], EfficientNet [94], ENet [85], ESPNet [38], CGNet [67], ESPNetv2 [39], EDANet [68], and LEDNet [69]. Among them, Inf-Net [25] is designed for COVID-19 infected area segmentation. U-Net [55], UNet++ [56], and Attention U-Net [57] are well-known models for medical image segmentation. Besides, MobileNet [29], MobileNetv2 [30], ShuffleNet [31], ShuffleNetv2 [32], and EfficientNet [94] are designed for lightweight image

TABLE XI
COMPARISON BETWEEN MINISEG AND SOME COMPETITIVE SEGMENTATION METHODS WITH THEIR OPTIMAL TRAINING SETTINGS.

Method	Metrics (%) on COVID-19-CT100					Metrics (%) on COVID-19-P9					Metrics (%) on COVID-19-P20					Metrics (%) on COVID-19-P1110				
	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓	mIoU ↑	SEN ↑	SPC ↑	DSC ↑	HD ↓
U-Net [55]	78.25	73.78	97.81	69.58	94.21	79.64	89.52	98.72	70.35	108.29	83.42	83.88	98.55	72.69	58.14	76.58	80.49	97.04	59.63	78.25
PSPNet [37]	78.01	71.77	97.34	66.73	96.51	83.96	87.08	99.11	76.78	70.58	82.30	83.85	99.00	72.16	61.50	72.98	80.22	98.47	56.05	79.83
DeepLabv3 [34]	81.75	84.18	97.57	75.39	79.95	83.19	86.33	98.97	75.69	68.91	82.13	82.38	98.26	73.04	57.64	74.89	**80.64**	**98.76**	58.76	81.35
DUpsampling [91]	81.95	84.94	97.71	75.56	83.30	82.59	76.34	97.85	72.28	60.29	83.86	81.04	97.92	73.86	56.69	74.23	66.28	93.05	54.71	71.99
Inf-Net [25]	**82.36**	77.69	**98.97**	75.03	84.86	81.46	78.93	99.16	72.39	65.71	67.84	71.30	98.79	66.27	68.49	76.48	63.49	95.83	58.02	**69.95**
MobileNet [29]	81.31	82.06	96.17	65.19	92.63	82.17	86.35	99.24	76.28	97.06	82.85	83.94	98.13	73.85	67.59	75.46	80.20	97.69	60.42	90.78
ShuffleNetv2 [32]	79.37	81.46	97.65	71.86	82.40	81.15	87.27	**99.31**	72.99	110.82	82.51	81.76	98.61	72.09	64.32	75.45	76.97	95.85	58.96	76.33
ENet [85]	79.77	81.98	97.60	72.34	90.85	81.04	82.46	98.93	72.15	89.62	78.01	75.66	97.25	68.85	65.09	75.93	76.42	96.99	59.02	81.86
CGNet [67]	80.25	82.19	97.04	72.56	80.83	77.04	73.45	93.19	62.71	122.93	83.11	81.96	97.72	73.89	52.54	75.37	75.14	96.96	57.04	74.12
EDANet [68]	80.22	83.96	97.31	72.85	81.14	82.37	85.46	99.05	74.12	66.59	81.12	77.28	97.42	70.89	65.83	74.57	74.39	96.68	56.24	79.31
MiniSeg	82.15	**84.95**	97.72	**75.91**	**74.42**	**85.31**	**90.60**	99.15	**80.06**	**58.46**	**84.49**	**85.06**	**99.05**	**76.27**	**51.06**	**78.33**	79.62	97.71	**64.84**	71.69

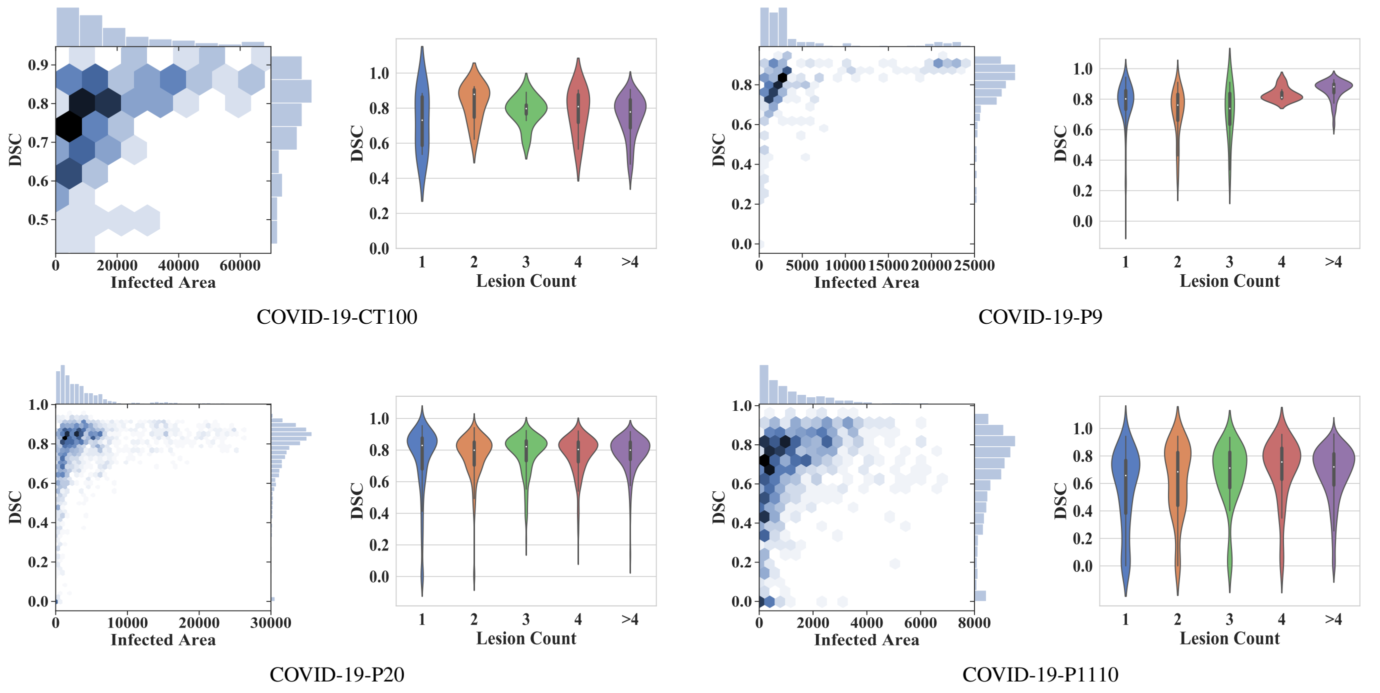

Fig. 4. Statistical analyses for MiniSeg on four datasets. Left of each sub-figure: The DSC score *vs.* the infected area; Right of each sub-figure: The DSC score *vs.* the lesion count in the corresponding CT slice.

classification. We view them as the encoder and add the decoder of MiniSeg to them so that they are reformed as image segmentation models. Moreover, ENet [85], ESPNet [38], CGNet [67], ESPNetv2 [39], EDANet [68], and LEDNet [69] are well-known lightweight segmentation models. The code of these methods is provided online by the authors. We believe that this benchmark would be useful for future research on COVID-19 segmentation.

The comparison between MiniSeg and other competitors, in terms of the number of parameters, the number of FLOPs, and speed, is summarized in Table X. We can clearly see that the numbers of parameters and FLOPs of MiniSeg are extremely small. Meanwhile, the speed of MiniSeg is much faster than others. Table X also provides the initialization information of all models. "Yes" indicates the models are initialized with the weights pre-trained on the ImageNet dataset [95], and "No" otherwise. Note that pre-training can be viewed as a pre-processing step, which needs to occupy additional training resources. It is clear that MiniSeg does not need to be pretrained on ImageNet [95] owing to its small model size, but can achieve superior performance. The numerical evaluation results of MiniSeg and other competitors are presented in Table IX. MiniSeg consistently achieves the best or close to the best performance in terms of all metrics on all datasets. For the metric of SPC, MiniSeg performs slightly worse than the best method on COVID-19-CT100 and COVID-19-P9. On the COVID-19-P1110 dataset, MiniSeg does not achieve the best results in terms of SEN. The fact that MiniSeg consistently outperforms other competitors demonstrates its effectiveness and superiority in COVID-19 infected area segmentation. Note that MiniSeg does not need to be pretrained on ImageNet [95] owing to its small model size. Therefore, we can come to the conclusion that MiniSeg has a low computational load, a fast speed, and good accuracy, making it convenient for practical deployment that is of high importance in the current severe situation of COVID-19.

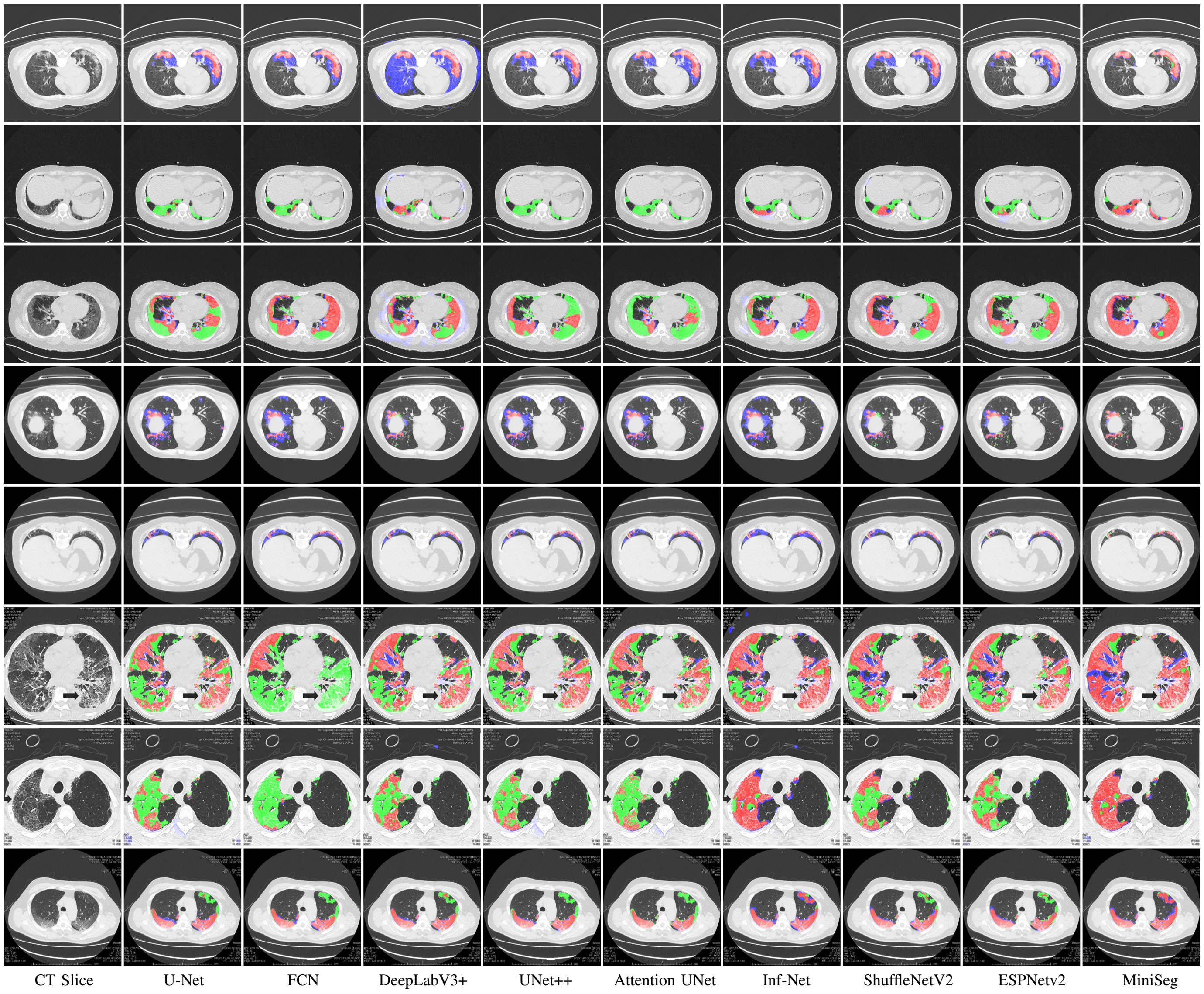

Fig. 5. Visual comparison between MiniSeg and other methods. Red: true positive; Green: false negative; Blue: false positive.

We can see that lightweight models, including Mobilenets [29], [30], ShuffleNets [31], [32], EfficientNet [94], and lightweight segmentation models, seem to outperform traditional segmentation networks, which validates our hypothesis that networks with a small number of parameters are more suitable for COVID-19 segmentation due to the limited training data. However, the proposed MiniSeg are more stable and superior across various evaluation metrics. Among traditional large networks, U-Net [55] and FCN-8s [54] achieve pretty good performance. This may be because the simple U-Net and FCN-8s are easy to train than other complicated networks, further demonstrating that previous cumbersome segmentation networks are not suitable for COVID-19 segmentation.

Moreover, we use the same training settings as MiniSeg to train all baseline models. However, different models may prefer different settings. Hence, to achieve a fair and rigorous comparison, we train multiple times for each baseline model with different training settings so that every model has a chance to find its optimal settings. Here, we mainly focus on four training hyper-parameters, *i.e.,* the learning rate, batch size, training epochs, and weight decay. In addition, this comparison is conducted only for some well-known or competitive baseline models. The results are shown in Table XI. As can be seen, the performance of baseline models is generally improved but still far worse than the proposed MiniSeg.

2) Statistical Analyses: To further study the characteristics of MiniSeg, we perform statistical analysis on COVID-19-CT100, COVID-19-P9, COVID-19-P20, and COVID-19-P1110 datasets. Fig. 4 illustrates the relationship between the DSC score and the size of the infected area or the lesion count in a CT slice. We find that MiniSeg achieves a DSC score larger than 0.7 for most CT slices regardless of the size of the infected area. The medium DSC is above 0.8 regardless of the lesion count. Only COVID-19-P1110 dataset has a small part of CT slices with results below 0.5. This suggests that MiniSeg is robust to different cases for COVID-19 infected area segmentation.

3) Qualitative Comparison: To explicitly show the superiority of MiniSeg, we provide a qualitative comparison between MiniSeg and eight previous state-of-the-art methods in Fig. 5. We select some representative images from the four datasets. This visual comparison further indicates that MiniSeg outperforms baseline methods remarkably. We can observe that the results of MiniSeg are very close to the ground-truth segmentation. Specifically, in the subtle regions, we can see that our method can successfully segment the COVID-19 infected areas with fine-grained details, leading to better infection segmentation.

V. Conclusion

Lots of recent studies show that the early screening of the infected areas is important to the fight against COVID-19. If the infected areas in radiological images can be segmented at the early stage, the patients will have a higher chance to survive. Hence, in this paper, we focus on segmenting COVID-19 infected areas from chest CT slices. To address the lack of COVID-19 training data and meet the efficiency requirement for the deployment of computer-aided COVID-19 screening systems, we propose an extremely minimum network, *i.e.,* MiniSeg, for accurate and efficient COVID-19 segmentation. MiniSeg adopts a novel multiscale learning module, *i.e.,* the **A**ttentive **H**ierarchical **S**patial **P**yramid (**AHSP**) module, to ensure its accuracy under the constraint of the extremely minimum network size. MiniSeg also utilizes a two-path architecture for learning complementary contextual and fine-grained features. To extensively compare MiniSeg with previous image segmentation methods and promote future research on COVID-19 segmentation, we build a comprehensive benchmark that would be useful for future research. The comparison between MiniSeg and state-of-the-art segmentation methods demonstrates that MiniSeg not only achieves the best efficacy but also has high efficiency. In conclusion, we provide an effective tool to assist radiologists to accurately determine the exact location and shape of the infected areas of COVID-19. Our study sheds some light on that artificial intelligence, especially deep learning, can thoroughly facilitate the screening and treatment of COVID-19.

In the future, we will further pay the effort to improve computer-aided COVID-19 screening sensitivity and make the network more lightweight, pushing this field to clinical deployment. Moreover, we will generalize the proposed MiniSeg into a wider range of practical scenarios, such as facilitating segmenting more types of lesions from radiological images.

References

[1] O. Tutsoy, "Pharmacological, non-pharmacological policies and mutation: An artificial intelligence based multi-dimensional policy making algorithm for controlling the casualties of the pandemic diseases," *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 2021.

[2] O. Tutsoy, Ş. Çolak, A. Polat, and K. Balikci, "A novel parametric model for the prediction and analysis of the COVID-19 casualties," *IEEE Access*, vol. 8, pp. 193 898–193 906, 2020.

[3] O. Tutsoy and A. Polat, "Linear and non-linear dynamics of the epidemics: System identification based parametric prediction models for the pandemic outbreaks," *ISA Transactions*, 2021.

[4] O. Tutsoy, K. Balikci, and N. F. Ozdil, "Unknown uncertainties in the COVID-19 pandemic: Multi-dimensional identification and mathematical modelling for the analysis and estimation of the casualties," *Digital Signal Process. (DSP)*, vol. 114, p. 103058, 2021.

[5] W. Wang, Y. Xu, R. Gao, R. Lu, K. Han, G. Wu, and W. Tan, "Detection of SARS-CoV-2 in different types of clinical specimens," *J. American Medical Association*, 2020.

[6] T. Ai, Z. Yang, H. Hou, C. Zhan, C. Chen, W. Lv, Q. Tao, Z. Sun, and L. Xia, "Correlation of chest CT and RT-PCR testing in coronavirus disease 2019 (COVID-19) in China: A report of 1014 cases," *Radiology*, vol. 296, no. 2, pp. E32–E40, 2020.

[7] Y. Fang, H. Zhang, J. Xie, M. Lin, L. Ying, P. Pang, and W. Ji, "Sensitivity of chest CT for COVID-19: Comparison to RT-PCR," *Radiology*, vol. 296, no. 2, pp. E115–E117, 2020.

[8] A. Shamsi, H. Asgharnezhad, S. S. Jokandan, A. Khosravi, P. M. Kebria, D. Nahavandi, S. Nahavandi, and D. Srinivasan, "An uncertainty-aware transfer learning-based framework for COVID-19 diagnosis," *IEEE Trans. Neur. Net. Learn. Syst. (TNNLS)*, vol. 32, no. 4, pp. 1408–1417, 2021.

[9] S. Dong, Q. Yang, Y. Fu, M. Tian, and C. Zhuo, "RCoNet: Deformable mutual information maximization and high-order uncertainty-aware learning for robust COVID-19 detection," *IEEE Trans. Neur. Net. Learn. Syst. (TNNLS)*, 2021.

[10] M. Yamaç, M. Ahishali, A. Degerli, S. Kiranyaz, M. E. Chowdhury, and M. Gabbouj, "Convolutional sparse support estimator-based COVID-19 recognition from X-ray images," *IEEE Trans. Neur. Net. Learn. Syst. (TNNLS)*, vol. 32, no. 5, pp. 1810–1820, 2021.

[11] N. Paluru, A. Dayal, H. B. Jenssen, T. Sakinis, L. R. Cenkeramaddi, J. Prakash, and P. K. Yalavarthy, "Anam-Net: Anamorphic depth embedding-based lightweight CNN for segmentation of anomalies in COVID-19 chest CT images," *IEEE Trans. Neur. Net. Learn. Syst. (TNNLS)*, vol. 32, no. 3, pp. 932–946, 2021.

[12] Y. Wang, H. Hou, W. Wang, and W. Wang, "Combination of CT and RT-PCR in the screening or diagnosis of COVID-19," *J. Global Health*, vol. 10, no. 1, p. 010347, 2020.

[13] M.-Y. Ng, E. Y. Lee, J. Yang, F. Yang, X. Li, H. Wang, M. M.-s. Lui, C. S.-Y. Lo, B. Leung, P.-L. Khong *et al.*, "Imaging profile of the COVID-19 infection: Radiologic findings and literature review," *Radiology: Cardiothoracic Imaging*, vol. 2, no. 1, p. e200034, 2020.

[14] S. Inui, A. Fujikawa, M. Jitsu, N. Kunishima, S. Watanabe, Y. Suzuki, S. Umeda, and Y. Uwabe, "Chest CT findings in cases from the cruise ship "Diamond Princess" with coronavirus disease 2019 (COVID-19)," *Radiology: Cardiothoracic Imaging*, vol. 2, no. 2, p. e200110, 2020.

[15] L. Li, L. Qin, Z. Xu, Y. Yin, X. Wang, B. Kong, J. Bai, Y. Lu, Z. Fang, Q. Song *et al.*, "Artificial intelligence distinguishes COVID-19 from community acquired pneumonia on chest CT," *Radiology*, vol. 296, no. 2, pp. E65–E71, 2020.

[16] F. Shi, J. Wang, J. Shi, Z. Wu, Q. Wang, Z. Tang, K. He, Y. Shi, and D. Shen, "Review of artificial intelligence techniques in imaging data acquisition, segmentation, and diagnosis for COVID-19," *IEEE Rev. in Bio. Eng. (RBME)*, vol. 14, pp. 4–15, 2020.

[17] D. S. W. Ting, L. Carin, V. Dzau, and T. Y. Wong, "Digital technology and COVID-19," *Nature Medicine*, vol. 26, no. 4, pp. 459–461, 2020.

[18] Y. Liu, Y.-H. Wu, Y. Ban, H. Wang, and M.-M. Cheng, "Rethinking computer-aided tuberculosis diagnosis," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2020, pp. 2646–2655.

[19] G. Liu, Y. Liao, F. Wang, B. Zhang, L. Zhang, X. Liang, X. Wan, S. Li, Z. Li, S. Zhang *et al.*, "Medical-VLBERT: Medical visual language BERT for COVID-19 CT report generation with alternate learning," *IEEE Trans. Neur. Net. Learn. Syst. (TNNLS)*, vol. 32, no. 9, pp. 3786–3797, 2021.

[20] C. Y. Lee and Y.-P. P. Chen, "New insights into drug repurposing for COVID-19 using deep learning," *IEEE Trans. Neur. Net. Learn. Syst. (TNNLS)*, 2021.

[21] Y. Liu, M.-M. Cheng, X. Hu, J.-W. Bian, L. Zhang, X. Bai, and J. Tang, "Richer convolutional features for edge detection," *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, vol. 41, no. 8, pp. 1939–1946, 2019.

[22] J. Mei, M.-M. Cheng, G. Xu, L.-R. Wan, and H. Zhang, "Sanet: A slice-aware network for pulmonary nodule detection," *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, 2021.

[23] M. Irfan, M. A. Iftikhar, S. Yasin, U. Draz, T. Ali, S. Hussain, S. Bukhari, A. S. Alwadie, S. Rahman, A. Glowacz *et al.*, "Role of hybrid deep neural networks (HDNNs), computed tomography, and chest X-rays for the detection of COVID-19," *Int. J. of Envir. Res. and Pub. Health (IJERPH)*, vol. 18, no. 6, p. 3056, 2021.

[24] Y. E. Almalki, A. Qayyum, M. Irfan, N. Haider, A. Glowacz, F. M. Alshehri, S. K. Alduraibi, K. Alshamrani, M. A. Alkhalik Basha,

A. Alduraibi *et al.*, "A novel method for COVID-19 diagnosis using artificial intelligence in chest X-ray images," in *Healthcare*, vol. 9, no. 5. Multidisciplinary Digital Publishing Institute, 2021, p. 522.

[25] D.-P. Fan, T. Zhou, G.-P. Ji, Y. Zhou, G. Chen, H. Fu, J. Shen, and L. Shao, "Inf-Net: Automatic COVID-19 lung infection segmentation from CT images," *IEEE Trans. Med. Imaging (TMI)*, vol. 39, no. 8, pp. 2626–2637, 2020.

[26] X. Chen, L. Yao, and Y. Zhang, "Residual attention U-Net for automated multi-class segmentation of COVID-19 chest CT images," *arXiv preprint arXiv:2004.05645*, 2020.

[27] A. Amyar, R. Modzelewski, H. Li, and S. Ruan, "Multi-task deep learning based CT imaging analysis for COVID-19 pneumonia: Classification and segmentation," *Comput. in Bio. and Med. (CIBM)*, vol. 126, p. 104037, 2020.

[28] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 1251–1258.

[29] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[30] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018, pp. 4510–4520.

[31] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018, pp. 6848–6856.

[32] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "ShuffleNet v2: Practical guidelines for efficient CNN architecture design," in *Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 116–131.

[33] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, vol. 40, no. 4, pp. 834–848, 2018.

[34] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[35] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 801–818.

[36] M. Yang, K. Yu, C. Zhang, Z. Li, and K. Yang, "DenseASPP for semantic segmentation in street scenes," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018, pp. 3684–3692.

[37] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 2881–2890.

[38] S. Mehta, M. Rastegari, A. Caspi, L. Shapiro, and H. Hajishirzi, "ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation," in *Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 552–568.

[39] S. Mehta, M. Rastegari, L. Shapiro, and H. Hajishirzi, "ESPNetv2: A light-weight, power efficient, and general purpose convolutional neural network," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019, pp. 9190–9200.

[40] T. Pohlen, A. Hermans, M. Mathias, and B. Leibe, "Full-resolution residual networks for semantic segmentation in street scenes," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 4151–4160.

[41] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Learning a discriminative feature network for semantic segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018, pp. 1857–1866.

[42] Y. Qiu, Y. Liu, S. Li, and J. Xu, "MiniSeg: An extremely minimum network for efficient COVID-19 segmentation," in *AAAI Conf. Artif. Intell. (AAAI)*, 2021, pp. 4846–4854.

[43] J. P. Cohen, P. Morrison, L. Dao, K. Roth, T. Q. Duong, and M. Ghassemi, "COVID-19 image data collection: Prospective predictions are the future," *arXiv preprint arXiv:2006.11988*, 2020.

[44] J. Zhao, Y. Zhang, X. He, and P. Xie, "COVID-CT-Dataset: A CT scan dataset about COVID-19," *arXiv preprint arXiv:2003.13865*, 2020.

[45] S. f. M. Italian, I. Radiology, and MILA, "Italian Society of Medical and Interventional Radiology COVID-19 Dataset," https://www.sirm.org/category/senza-categoria/covid-19/, 2020.

[46] M. de la Iglesia Vayá, J. M. Saborit, J. A. Montell, A. Pertusa, A. Bustos, M. Cazorla, J. Galant, X. Barber, D. Orozco-Beltrán, F. García-García *et al.*, "BIMCV COVID-19+: A large annotated dataset of RX and CT images from COVID-19 patients," *arXiv preprint arXiv:2006.01174*, 2020.

[47] M. E. Chowdhury, T. Rahman, A. Khandakar, R. Mazhar, M. A. Kadir, Z. B. Mahbub, K. R. Islam, M. S. Khan, A. Iqbal, N. Al Emadi *et al.*, "Can AI help in screening viral and COVID-19 pneumonia?" *IEEE Access*, vol. 8, pp. 132 665–132 676, 2020.

[48] J. Born, G. Brändle, M. Cossio, M. Disdier, J. Goulet, J. Roulin, and N. Wiedemann, "POCOVID-Net: Automatic detection of COVID-19 from a new lung ultrasound imaging dataset (POCUS)," *arXiv preprint arXiv:2004.12084*, 2020.

[49] E. Soares, P. Angelov, S. Biaso, M. H. Froes, and D. K. Abe, "SARS-CoV-2 CT-scan dataset: A large dataset of real patients CT scans for SARS-CoV-2 identification," *medRxiv*, 2020.

[50] A. M. Dadário, "COVID-19 X-rays," https://www.kaggle.com/andrewmvd/convid19-X-rays, 2020.

[51] H. B. Jenssen, "COVID-19 CT Segmentation Dataset," http://medicalsegmentation.com/covid19/, 2020.

[52] M. Jun, G. Cheng, W. Yixin, A. Xingle, G. Jiantao, Y. Ziqi, Z. Minqing, L. Xin, D. Xueyuan, C. Shucheng *et al.*, "COVID-19 CT Lung and Infection Segmentation Dataset," *Zenodo, Apr*, vol. 20, 2020.

[53] S. Morozov, A. Andreychenko, N. Pavlov, A. Vladzymyrskyy, N. Ledikhova, V. Gombolevskiy, I. Blokhin, P. Gelezhe, A. Gonchar, V. Chernina *et al.*, "MosMedData: Chest CT scans with COVID-19 related findings," *medRxiv*, 2020.

[54] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, vol. 39, no. 4, pp. 640–651, 2017.

[55] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Int. Conf. Med. Image Comp. Comput.-Assist. Interv. (MICCAI)*, 2015, pp. 234–241.

[56] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "UNet++: A nested U-Net architecture for medical image segmentation," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer, 2018, pp. 3–11.

[57] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz *et al.*, "Attention U-Net: Learning where to look for the pancreas," in *Medical Imaging with Deep Learning*, 2018.

[58] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Int. Conf. Comput. Vis. (ICCV)*, 2015, pp. 1520–1528.

[59] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, vol. 39, no. 12, pp. 2481–2495, 2017.

[60] H. Zhang, K. Dana, J. Shi, Z. Zhang, X. Wang, A. Tyagi, and A. Agrawal, "Context encoding for semantic segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2018, pp. 7151–7160.

[61] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "CCNet: Criss-cross attention for semantic segmentation," in *Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 603–612.

[62] Z. Zhu, M. Xu, S. Bai, T. Huang, and X. Bai, "Asymmetric non-local neural networks for semantic segmentation," in *Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 593–602.

[63] Z. Wu, C. Shen, and A. v. d. Hengel, "Wider or deeper: Revisiting the resnet model for visual recognition," *Pattern Recogn.*, vol. 90, pp. 119–133, 2019.

[64] J. Jin, A. Dundar, and E. Culurciello, "Flattened convolutional neural networks for feedforward acceleration," in *Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–11.

[65] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in *AAAI Conf. Artif. Intell. (AAAI)*, 2017, pp. 4278–4284.

[66] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 1492–1500.

[67] T. Wu, S. Tang, R. Zhang, and Y. Zhang, "CGNet: A light-weight context guided network for semantic segmentation," *IEEE Trans. Image Process. (TIP)*, vol. 30, pp. 1169–1179, 2020.

[68] S.-Y. Lo, H.-M. Hang, S.-W. Chan, and J.-J. Lin, "Efficient dense modules of asymmetric convolution for real-time semantic segmentation," in *ACM Multimedia Asia*, 2019, pp. 1–6.

[69] Y. Wang, Q. Zhou, J. Liu, J. Xiong, G. Gao, X. Wu, and L. J. Latecki, "LEDNet: A lightweight encoder-decoder network for real-time semantic segmentation," in *Int. Conf. Image Process. (ICIP)*. IEEE, 2019, pp. 1860–1864.

[70] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia, "ICNet for real-time semantic segmentation on high-resolution images," in *Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 405–420.

[71] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2016, pp. 4820–4828.
[72] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in *Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–14.
[73] W. Chen, J. T. Wilson, S. Tyree, K. Q. Weinberger, and Y. Chen, "Compressing neural networks with the hashing trick," in *Int. Conf. Mach. Learn. (ICML)*, 2015, pp. 2285–2294.
[74] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," *J. Mach. Learn. Res. (JMLR)*, vol. 18, no. 1, pp. 6869–6898, 2017.
[75] H. Bagherinezhad, M. Rastegari, and A. Farhadi, "LCNN: Lookup-based convolutional neural network," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 7120–7129.
[76] B. Liu, M. Wang, H. Foroosh, M. Tappen, and M. Pensky, "Sparse convolutional neural networks," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2015, pp. 806–814.
[77] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *Annu. Conf. Neur. Inform. Process. Syst. (NeurIPS)*, 2016, pp. 2074–2082.
[78] A. Narin, C. Kaya, and Z. Pamuk, "Automatic detection of coronavirus disease (COVID-19) using X-ray images and deep convolutional neural networks," *Pattern Analysis and Applications*, vol. 24, no. 3, pp. 1207–1220, 2021.
[79] O. Gozes, M. Frid-Adar, H. Greenspan, P. D. Browning, H. Zhang, W. Ji, A. Bernheim, and E. Siegel, "Rapid AI development cycle for the coronavirus (COVID-19) pandemic: Initial results for automated detection & patient monitoring using deep learning CT image analysis," *arXiv preprint arXiv:2003.05037*, 2020.
[80] X. Xu, X. Jiang, C. Ma, P. Du, X. Li, S. Lv, L. Yu, Q. Ni, Y. Chen, J. Su *et al.*, "A deep learning system to screen novel coronavirus disease 2019 pneumonia," *Engineering*, vol. 6, no. 10, pp. 1122–1129, 2020.
[81] J. Zhang, Y. Xie, Y. Li, C. Shen, and Y. Xia, "COVID-19 screening on chest X-ray images using deep learning based anomaly detection," *arXiv preprint arXiv:2003.12338*, 2020.
[82] L. Wang, Z. Q. Lin, and A. Wong, "COVID-Net: A tailored deep convolutional neural network design for detection of COVID-19 cases from chest X-ray images," *Scientific Reports*, vol. 10, no. 1, pp. 1–12, 2020.
[83] A. Abbas, M. M. Abdelsamea, and M. M. Gaber, "4S-DT: Self-supervised super sample decomposition for transfer learning with application to COVID-19 detection," *IEEE Trans. Neur. Net. Learn. Syst. (TNNLS)*, 2021.
[84] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Int. Conf. Comput. Vis. (ICCV)*, 2015, pp. 1026–1034.
[85] A. Paszke, A. Chaurasia, S. Kim, and E. Culurciello, "ENet: A deep neural network architecture for real-time semantic segmentation," *arXiv preprint arXiv:1606.02147*, 2016.
[86] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017, pp. 4700–4708.
[87] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *Annu. Conf. Neur. Inform. Process. Syst. (NeurIPS)*, 2019, pp. 8026–8037.
[88] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–15.
[89] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 325–341.
[90] Y. Yuan, L. Huang, J. Guo, C. Zhang, X. Chen, and J. Wang, "OCNet: Object context for semantic segmentation," *Int. J. Comput. Vis. (IJCV)*, vol. 129, no. 8, pp. 2375–2398, 2021.
[91] Z. Tian, T. He, C. Shen, and Y. Yan, "Decoders matter for semantic segmentation: Data-dependent decoding enables flexible feature aggregation," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019, pp. 3126–3135.
[92] J. Fu, J. Liu, H. Tian, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *IEEE Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2019, pp. 3146–3154.
[93] X. Li, H. Zhao, L. Han, Y. Tong, S. Tan, and K. Yang, "Gated fully fusion for semantic segmentation," in *AAAI Conf. Artif. Intell. (AAAI)*, 2020, pp. 11 418–11 425.
[94] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 6105–6114.
[95] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis. (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

Yu Qiu is a PhD candidate at the College of Artificial Intelligence, Nankai University. She received her bachelor's degree from Northwest A&F University in 2017. Her research interests include computer vision and machine learning.

Yun Liu received his bachelor's and doctoral degrees from Nankai University in 2016 and 2020, respectively. Currently, he works with Prof. Luc Van Gool as a postdoctoral scholar at Computer Vision Lab, ETH Zurich. His research interests include computer vision and machine learning.

Shijie Li received his bachelor's degree in Automation Engineering from University of Electronic Science and Technology of China in 2016 and his master's degree in computer science from Nankai University in 2019. Since 2019, he is a Ph.D. student at the University of Bonn. His research interests include action recognition and scene understanding.

Jing Xu is a professor at the College of Artificial Intelligence, Nankai University. She received her doctoral degree from Nankai University in 2003. She has published more than 100 papers in software engineering, software security, and big data analytics. She won the second prize of Tianjin Science and Technology Progress Award twice in 2017 and 2018, respectively.